MINISTRY OF EDUCATION
AND TRAINING

VIETNAM ACADEMY
OF SCIENCE AND TECHNOLOG

GRADUATE UNIVERSITY OF SCIENCE AND
TECHNOLOGY

Đào Thị Trang

MAXIMUM ANTICHAIN PROBLEM AND AN APPLICATION IN COP-ROBBER GUARDING GAME

MASTER THESIS IN APPLIED MATHMATICS

Ha Noi – 2023

MINISTRY OF EDUCATION AND TRAINING
VIETNAM ACADEMY OF SCIENCE AND TECHNOLOGY

GRADUATE UNIVERSITY OF SCIENCE AND TECHNOLOGY

Đào Thị Trang

MAXIMUM ANTICHAIN PROBLEM AND AN APPLICATION IN COP-ROBBER GUARDING GAME

MASTER THESIS IN APPLIED MATHMATICS
Code: 8 46 01 12

ADVISORS:
1. TS. Lê Xuân Thanh
2. PGS. TS. Bùi Văn Định

Hà Nội - 2023

BỘ GIÁO DỤC
VÀ ĐÀO TẠO

VIỆN HÀN LÂM KHOA HỌC
VÀ CÔNG NGHỆ VIỆT NAM

HỌC VIỆN KHOA HỌC VÀ CÔNG NGHỆ

Đào Thị Trang

BÀI TOÁN TÌM ANTICHAIN CỰC ĐẠI VÀ ÁP DỤNG TRONG TRÒ CHƠI PHÒNG THỦ CẢNH SÁT – CƯỚP

LUẬN VĂN THẠC SĨ TOÁN ỨNG DỤNG
Mã số: 8 46 01 12

NGƯỜI HƯỚNG DẪN KHOA HỌC :
1. TS. Lê Xuân Thanh
2. PGS. TS. Bùi Văn Định

Lê Xuân Thanh Bùi Văn Định

Hà Nội - 2023

Commitment

This thesis is done by my own study under the supervision of Dr. Le Xuan Thanh and Assoc. Prof. Dr. Bui Van Dinh. It has not been defensed in any council and has not been published on any media. The results as well as the ideas of other authors are all specifically cited. I take full responsibility for my commitment.

Hanoi, September 2023

Dao Thi Trang

Acknowledgements

I would like to express my gratitude to the teachers in Institute of Mathematics and Graduate University of Science and Technology, Vietnam Academy of Science and Technology whose lectures have imparted valuable knowledge and facilitated me to complete the master courses and the thesis. In particular, I would like to express my deep respect gratitude to Dr. Le Xuan Thanh and Assoc. Prof. Dr. Bui Van Dinh, who have direct guidance and helped me to complete this thesis.

Due to limited time, capacity and conditions, the thesis cannot avoid errors. I look forward to receiving valuable comments from readers.

Hanoi, September 2023

Dao Thi Trang

Introduction

The central problem studied in this thesis is the maximum antichain problem, which is stated shortly as follows: "*Given a partially ordered set S of finite cardinality, find a subset $A \subseteq S$ having as many elements as possible such that any two distinct elements of A are incomparable*". This problem has an intuitive statement in the language of graph theory as follows: "*Given a simple directed acyclic graph G, find a vertex subset A in G having as many vertices as possible such that there is no directed path connecting any two distinct vertices of A*".

In 1950, Robert Palmer Dilworth - an American mathematician - stated a theorem in [1] about the size of a solution to the maximum antichain problem. The theorem was then named after Dilworth. Later, in 1956, George Bernard Dantzig and Alan Jerome Hoffman - two other American mathematicians - gave a proof for this theorem in [2]. Their proof bases on the well-known linear programming duality. In 1963, Micha Asher Perles - an Israeli mathematician - gave another proof in [3] for the theorem, which uses an elegant induction technique. *In Chapter 1 of this thesis, we will focus on the statement of Dilworth's theorem as well as the two mentioned proofs for this theorem.*

Dilworth's theorem gives the base line for constructing a polynomial time algorithm to solve the maximum antichain problem. Such an algorithm can be traced from the proof of Corollary 14.7b in [4]. This result plays an important role in studying applications of maximum antichain problem. Such an interesting application has been studied by Hiroshi Nagamochi in [5]. The paper considers a cop-robber guarding game played on an undirected graph, in which the robber region is a cycle, and the objective is to find a strategy with minimum number of cops to guard the vertices outside the robber region. By constructing an auxiliary graph, Nagamochi showed that the size of a maximum antichain in the auxiliary graph is the best lower bound for the optimal number of cops in the game. It is worth noting that, with some further technical arguments, Nagamochi proved in the same paper that such

the lower bound equals the optimal number of cops. *In Chapter 2, we will present the description of the cop-robber guarding game problem and the detail proof of the result of Nagamochi on the best lower bound for the optimal objective value of the problem.*

The contents of this thesis are organized in two chapters as discussed above. The thesis is closed with a summary in the conclusion part. It is worth noting that the main results in this thesis are not new. Our main contributions in the thesis consist of the followings.

- We give the detail proof for the graph version of Dilworth's theorem in Section 1.2. As far as we known, this theorem's version is stated and proved very briefly in Chapter 14 of [4]. In Section 1.2 we translate the proof of Dantzig and Hoffman in [2], which is for the poset version of the theorem, to the proof for the graph version.
- In Section 1.3 we translate the induction proof of Perles [3], which is for the poset version of Dilworth's theorem, to a detail proof for the graph version of the theorem.
- We explain in detail the construction of Nagamochi in [5] for the problem studied in Chapter 2, and provide an example that is different from the one in [5].

Chapter 1

Dilworth's theorem

The research object in the center of this chapter is Dilworth's theorem, which is an important result on the maximum antichain problem. We recall in Section 1.1 some preliminaries and give detail statements for two equivalent versions of the maximum antichain problem as well as Dilworth's theorem. Section 1.2 and Section 1.3 respectively present two proofs for the theorem: one of Dantzig and Hoffmann in [2], the other of Perles in [3].

1.1 Theorem statement

Both the maximum antichain problem and Dilworth's theorem have two versions: one for posets and the other for acyclic simple digraphs. In this section, we first give the detail statements for their poset version, then present their graphical version.

1.1.1 Poset version

In the sequel, X is a finite set.

Definition 1.1. (Binary relation, see e.g. [6], Chapter 3). *A binary relation on X is a subset of $X \times X$.*

Let $E \subseteq X \times X$. A binary relation on X defined by the subset E can be seen as a mapping $R : X \times X \to \{0, 1\}$ in which

$$R(x, y) = \begin{cases} 1 & \text{if } (x, y) \in E, \\ 0 & \text{if } (x, y) \notin E. \end{cases}$$

We also write xRy to indicate that $R(x, y) = 1$.

Definition 1.2. (Partial order, see e.g. [4], Chapter 14). *A partial order on X is a binary relation $\preccurlyeq$ on X satisfying the following properties:*

(i) Reflexivity: $a \preccurlyeq a$ for every $a \in X$;

(ii) Antisymmetry: For all $a, b \in X$, if $a \preccurlyeq b$ and $b \preccurlyeq a$, then $a = b$;

(iii) Transitivity: For all $a, b, c \in X$, if $a \preccurlyeq b$ and $b \preccurlyeq c$, then $a \preccurlyeq c$.

If $a \preccurlyeq b$, we say that a precedes b according to the relation $\preccurlyeq$. For $a, b \in X$, we say that they are comparable if either $a \preccurlyeq b$ or $b \preccurlyeq a$. The term 'partial' indicates that not every pair of elements in X needs to be comparable. We write $a \prec b$ if $a \preccurlyeq b$ and $a \neq b$.

Definition 1.3. (Poset, see e.g. [4], Chapter 14). *The set X is called a partially ordered set, or poset for short, if it is equipped by a partial order $\preccurlyeq$.*

Example 1.4. Consider the set $X = \{1, 2, 3, 4, 5, 6, 7\}$ and the subset $E \subset X \times X$ given by

$$\begin{aligned} E = \{&(1,1), (2,2), (3,3), (4,4), (5,5), (6,6), (7,7), (1,2), \\ &(1,3), (3,2), (2,4), (1,4), (4,7), (1,7), (2,7), (3,4), \\ &(3,5), (1,5), (5,7), (3,7), (3,6), (1,6), (6,7)\}. \end{aligned}$$

By checking exhaustively the reflexivity, antisymmetry, and transitivity properties described in Definition 1.2, the binary relation $\preccurlyeq$ defined by the set E is a partial order, and hence the set X together with this relation $\preccurlyeq$ is a poset.

Definition 1.5. (Chains and antichains in posets, see e.g. [4], Chapter 14). *Let $(X, \preccurlyeq)$ be a poset.*

(i) A chain in X is a subset $C \subseteq X$ such that for any pair $a, b \in C$ we have either $a \preccurlyeq b$ or $b \preccurlyeq a$.

(ii) An antichain in X is a subset $A \subseteq X$ such that for all $a, b \in A$ we have $a \not\preccurlyeq b$ and $b \not\preccurlyeq a$. The size of an antichain A is the number $|A|$ of its elements.

(iii) A maximum antichain in X is an antichain of maximum size.

Roughly speaking, a chain in X is a subset whose elements are pairwise comparable, while an antichain in X is a subset in which the elements are pairwise non-comparable. As an illustration, for the poset $(X, \preccurlyeq)$ in Example 1.4, the set $\{1, 3, 5, 7\}$ is a chain since $1 \prec 3 \prec 5 \prec 7$. This is because $\{(1,3), (1,5), (1,7), (3,5), (3,7), (5,7)\} \subset E$. Also in this example we have $\{2, 5\}$ is an antichain since $(2,5) \notin E$ and $(5,2) \notin E$, i.e., $2 \not\prec 5$ and $5 \not\prec 2$. This antichain has size 2 since it consists of two elements. By similar arguments,

$\{2, 5, 6\}$ is also an antichain in the example. This antichain has size 3, and in fact it is a maximum antichain of the poset $(X, \preccurlyeq)$.

The poset version of the maximum antichain problem reads as follows.

> "*Let $(X, \preccurlyeq)$ be a poset. Find a maximum antichain with respected to the order $\preccurlyeq$ in X.*"

Dilworth states in [1] a theorem about the size of a maximum antichain in a poset. For the statement of the theorem, we need the following additional concept.

Definition 1.6. (see [4], Chapter 14). *Let $(X, \preccurlyeq)$ be a poset, $Y \subseteq X$, and $\mathcal{C}$ a collection of chains in the poset. We say that $\mathcal{C}$ covers Y if each element in Y belongs to a chain in $\mathcal{C}$.*

We again take the poset $(X, \preccurlyeq)$ in Example 1.4 to illustrate this concept. There, let us consider the collection $\mathcal{C}$ consisting of two chains $\{1, 2, 4\}$ and $\{1, 3, 5, 7\}$. This collection obviously covers the set $Y = \{2, 5\}$, since the former (resp., the latter) element in Y belongs to the former (resp., the latter) chain in $\mathcal{C}$.

It is now ready to state Dilworth's theorem.

Theorem 1.7. (Dilworth's theorem for posets). *Let $(X, \preccurlyeq)$ be a poset. Then the minimum number of chains covering X equals the size of a maximum antichain in this poset.*

1.1.2 Graphical version

It is more intuitive for us when representing the concepts and results in the previous subsection in terms of graphs. Throughout this thesis, by graphs we mean simple ones, i.e., (i) there is no edge or arc whose endpoints are the same vertex and (ii) there are no pair of edges or arcs sharing common endpoints.

Given a poset $(X, \preccurlyeq)$, we construct a directed graph G associated with this poset as follows.

- The vertex set of G consists of elements of X.
- For any pair of distinct vertices a, b in G, a directed arc (a, b) is formed if $a \prec b$ and there does not exist a vertex $c \in X \backslash \{a, b\}$ such that $a \prec c \prec b$.

As an illustration, the graph associated with the poset in Example 1.4 is shown in Figure 1.1. In this way of construction, two distinct elements u and

v are comparable in the sense that $u \prec v$ if there is a directed path from u to v in G. Indeed, let $u - i_1 - i_2 - \ldots - i_k - v$ be such a path. Since (u, i_1) is an arc of G, we have $u \prec i_1$. Similarly, we have $i_1 \prec i_2 \prec \ldots \prec i_k \prec v$. By transitivity of the partial order $\preccurlyeq$, we have $u \prec v$. In addition, it is clear that the graph G constructed in this way is acyclic. Indeed, assume the contrary that there exists a cycle $v_1 - v_2 - \ldots - v_k - v_1$ in G with $v_k \neq v_1$. Since $v_1 - v_2 - \ldots - v_k$ is a path, we have $v_1 \preccurlyeq v_k$. Since (v_k, v_1) is an arc in the cycle, we have $v_k \preccurlyeq v_1$. Thus, by antisymmetry of the partial relation $\preccurlyeq$, we have $v_1 = v_k$, which contradicts our setting that $v_k \neq v_1$. This contradiction proves the acyclic property of G.

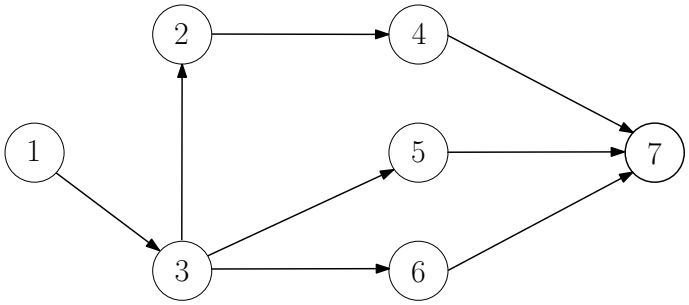

Figure 1.1: The graph associated with the poset in Example 1.4.

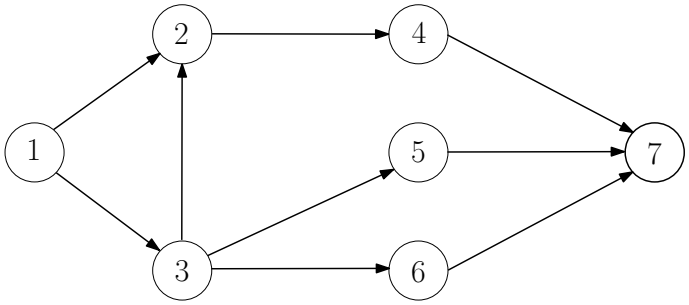

Figure 1.2: A directed acyclic graph that induces a partial order.

Conversely, given a directed acyclic graph G, we can construct a poset $(X, \preccurlyeq)$ associated with G as follows.

- The set X consists of the vertices of G.
- Each element of X is comparable to itself with respect to the relation $\preccurlyeq$.
- Two distinct elements $a, b \in X$ are comparable in the sense that $a \prec b$ if there is a directed path from a to b in G.

In this way of construction, the relation $\preccurlyeq$ is in fact a partial order. This comes from the following arguments.

- This relation is reflexive, since each element of X is comparable to itself with respect to $\preccurlyeq$.
- This relation is antisymmetric, since the graph G is acyclic. Indeed, let $a, b \in X$ be such that $a \preccurlyeq b$ and $b \preccurlyeq a$. If $a \neq b$, then there exists a directed path P from a to b (since $a \preccurlyeq b$) and there exists a directed path P' from b to a (since $b \preccurlyeq a$). The concatenation of P and P' form a cycle going through a and b, which contradicts the acyclic property of G. Therefore we must have $a = b$, which proves the antisymmetry of $\preccurlyeq$.
- This relation is transitive. Indeed, let $a, b, c \in X$ be pairwise distinct satisfying $a \preccurlyeq b$ and $b \preccurlyeq c$. Since $a \preccurlyeq b$, there exists a directed path P_1 from a to b in G. Similarly, since $b \preccurlyeq c$, there exists a directed path P_2 from b to c in G. The concatenation of P_1 and P_2 is a directed path from a to c, which means $a \preccurlyeq c$. This proves the transitivity of $\preccurlyeq$.

As an illustration, the graph in Figure 1.2 induces the poset in Example 1.4. By the correspondence between posets and their associated directed acyclic graphs as we have discussed, for each concept and result concerning posets we have a corresponding version in terms of graphs.

Definition 1.8. (Chains and antichains in directed acyclic graphs, see e.g. [4], Chapter 14). *Let G be a directed acyclic graph.*

(i) A chain in G is a directed path in this graph. If there is a chain from a vertex u to a vertex v in G, then the chain is also called a u-v-path, and we say that v is reachable from u, or u is reachable to v.

(ii) An antichain in G is a subset A of the vertex set of G such that there is no chain between any pair of distinct vertices in G. The size of an antichain A is the number $|A|$ of its elements.

(iii) A maximum antichain in G is an antichain A of maximum size.

To illustrate, for the graph in Figure 1.1, $\{2, 5\}$ is an antichain (since there is no directed path from vertex 2 to vertex 5 and vice versa). Similarly, each set $\{2, 6\}, \{4, 5\}, \{4, 6\}, \{5, 6\}$ is also an antichain in this graph. These antichains are of size 2. The sets $\{2, 5, 6\}$ and $\{4, 5, 6\}$ are also antichains of the same size 3, and they are maximum antichains in the graph. Hence, this example shows that the maximum antichain in a directed acyclic graph can be non-unique.

Definition 1.9. (see [4], Chapter 14). *Let G be a directed acyclic graph, B a subset of the vertex set of G, and $\mathcal{P}$ a set of paths in G.*

(i) We say that $\mathcal{P}$ covers B if each vertex in B belongs to a path in $\mathcal{P}$. In this case we also say that $\mathcal{P}$ is a path covering of B. The number of paths in $\mathcal{P}$, denoted $|\mathcal{P}|$, is called the size of the path covering $\mathcal{P}$.

(ii) We say that $\mathcal{P}$ is a minimum path covering of B if it covers B and for any path covering $\mathcal{P}'$ of B we have $|\mathcal{P}'| \geq |\mathcal{P}|$.

(iii) We say that $\mathcal{P}$ is a path covering of G if it covers the vertex set of this graph.

Roughly speaking, $\mathcal{P}$ is a minimum path covering of B if it has minimum size among the path coverings of B. For example, in the graph in Figure 1.2, the set $\mathcal{P}$ consisting of two paths $1-2-4$ and $1-3-5-7$ covers $B = \{2, 5\}$ but does not cover $B' = \{2, 5, 6\}$ (since the vertex 6 does not belong to any path in $\mathcal{P}$). Since there is no path from vertex 2 to vertex 5 and vice versa, the path covering $\mathcal{P}$ is also a minimum path covering of $B = \{2, 5\}$.

The maximum antichain problem can be now stated in terms of graphs as follows.

"*Find a maximum antichain in a given directed acyclic graph G.*"

The following theorem restates Theorem 1.7 in the language of graph theory.

Theorem 1.10. (Dilworth's theorem for acyclic digraphs). *Let G be a directed acyclic graph. Then the size of a minimum path covering of G equals the size of a maximum antichain in G.*

Given a directed acyclic graph G and $(V, \preccurlyeq)$ the poset associated with G, by applying Theorem 1.7 to the poset $(V, \preccurlyeq)$, we can view Theorem 1.10 as a corollary of the poset version of Dilworth's theorem. However, for intuition purpose, in the next two sections we will present two direct proofs for the graphical version of Dilworth's theorem (Theorem 1.10) using the language of graph theory.

1.2 First proof

Dantzig and Hoffman in [2] give a proof for the poset version of Dilworth's theorem based on the well-known linear programming duality. Following their proof, in this section we present a detail proof for the graphical version of Dilworth's theorem. It is worth noting that some concepts and results in this

section are introduced without citation, they are just technical concepts and results that are used for the convenience of translating the proof of Dantzig and Hoffman for poset version to the one for graph version.

To prove Theorem 1.10 on a given directed acyclic graph G, the following three problems are concerned.

- (P1) Find a minimum path covering of G.
- (P2) Find a minimum vertex-disjoint path covering of G in its transitive closure $\overline{G}$.
- (P3) Find a maximum antichain in G.

The proof we are going to show in this section can be sketched into three following steps.

- Step 1: Show that (P1) is equivalent to (P2).
- Step 2: Formulate (P2) as a linear program.
- Step 3: Show that the dual of (P2) gives an optimal solution for (P3).

We will discuss these steps respectively in the following three subsections.

1.2.1 Transitive closure graph

Let V be the vertex set and E the arc set of the given directed acyclic graph G. We construct a directed graph $\overline{G}$ associated with G as follows.

- The vertex set of $\overline{G}$ is also the vertex set V of G.
- For any pair of distinct vertices $u, v \in V$, we establish a directed arc (u, v) in $\overline{G}$ if there is a directed path from u to v in G.

Let $\overline{E}$ be the arc set of $\overline{G}$. This arc set has the following property.

Proposition 1.11. *Let $u, v, w \in V$ be pairwise distinct. If $(u, v) \in \overline{E}$ and $(v, w) \in \overline{E}$, then $(u, w) \in \overline{E}$.*

Proof. By construction of $\overline{G}$, since $(u, v) \in \overline{E}$, there is a directed path P_1 from u to v in G. Similarly, since $(v, w) \in \overline{E}$, there is a directed path P_2 from v to w in G. Let P be the concatenation of P_1 and P_2 at v, then P is a directed path from u to w. Then (u, w) is an arc in $\overline{G}$. □

The above proposition means that the transitivity between arcs holds in $\overline{G}$, although this property might not hold for the original graph G. Therefore, we

call $\overline{G}$ the *transitive closure* of G. As an illustration, the graph in Figure 1.3 is the transitive closure of the one in Figure 1.2, in which the dashed arcs are not in the original graph. The following proposition shows that an important property of G is preserved in its transitive closure.

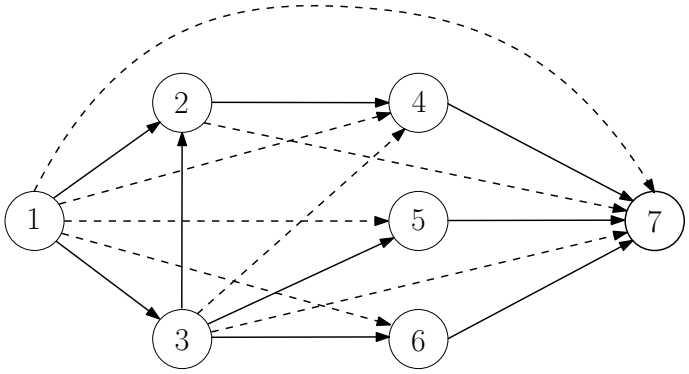

Figure 1.3: The transitive closure graph of the one in Figure 1.2.

Proposition 1.12. *The transitive closure graph $\overline{G}$ is also directed acyclic.*

Proof. By construction, $\overline{E}$ consists of directed arcs, hence $\overline{G}$ is directed. Assume the contrary that there exists a cycle $v_1 - v_2 - \ldots - v_k - v_1$ in $\overline{G}$. For each $i = 1, \ldots, k-1$, since (v_i, v_{i+1}) is an arc in $\overline{G}$, there exists a path P_i from v_i to v_{i+1} in the original graph G. Lastly, since (v_k, v_1) is an arc in $\overline{G}$, there is a path P_k from v_k to v_1 in G. By consecutively concatenating $P_1, \ldots, P_k$ we obtain a cycle P in G going through $v_1, \ldots, v_k$ and back to v_1. The existence of the cycle P contradicts the acyclic property of G. This contradiction implies that $\overline{G}$ is acyclic. □

For the discussion in the sequel, we need the following concepts.

Definition 1.13. *(i) A path covering $\mathcal{P}$ of G is called improper if there is a path $P \in \mathcal{P}$ such that each vertex of P belongs also to another path in $\mathcal{P}$.*

(ii) A path covering of G is called proper if it is not improper.

(iii) A path covering $\mathcal{P}$ of G is called vertex-disjoint if the paths in $\mathcal{P}$ are pairwise vertex-disjoint (i.e., there is no common vertex between any pair of paths in $\mathcal{P}$).

(iv) A path consisting of a single vertex is called a degenerated path.

If $\mathcal{P}$ is an improper path covering of G, then by definition there exists a path $P \in \mathcal{P}$ whose vertices are covered by the other paths in $\mathcal{P}$. In this case we can remove the path P from $\mathcal{P}$ and obtain a path covering $\mathcal{P}' = \mathcal{P} \backslash \{P\}$ that

still covers the vertex set of G. Therefore, without loss of generality, we can assume that **all path coverings under our consideration are proper**. Furthermore, it is worth noting that every minimum path covering must be proper.

The following lemma is important for proving the main result in this subsection.

Lemma 1.14. *Let $\mathcal{P}$ be a path covering of G. Then there exists a vertex-disjoint path covering $\overline{\mathcal{P}}$ of $\overline{G}$ having the same size as $\mathcal{P}$.*

If $\mathcal{P}$ consists of only one path P, then we simply take $\overline{\mathcal{P}} = \{P\}$. Therefore we only need to consider the case that $|\mathcal{P}| \geq 2$. We will prove this lemma by giving an algorithm to construct $\overline{\mathcal{P}}$ from $\mathcal{P}$. The key idea of this algorithm is to add each path of $\mathcal{P}$ into $\overline{\mathcal{P}}$ until an intersection by vertex appears, then going through a transitive closure arc to avoid the intersection vertex on the current path. Here, we say that two paths intersect if they have a common vertex, and such vertex is called an intersection one. The precise description of the algorithm is given below.

Algorithm 1 Construct a vertex-disjoint path covering of $\overline{G}$ from a given path covering of G

1: **Input:** A path covering $\mathcal{P} = \{P_1, \ldots, P_p\}$ of G with $p \geq 2$.
2: **Output:** A vertex-disjoint path covering $\overline{\mathcal{P}}$ of $\overline{G}$ such that $|\overline{\mathcal{P}}| = |\mathcal{P}|$.
3: **for** $i = 1, \ldots, p-1$ **do**
4: **if** P_i has only one vertex, say v, that is not in the sequel paths $P_{i+1}, \ldots, P_p$ **then**
5: $\overline{P_i} := \{v\}$
6: **else**
7: Let $v_1^i, \ldots, v_{\ell_i}^i$ be the vertices in P_i (respectively appear in the direction of P_i) that are not in the sequel paths $P_{i+1}, \ldots, P_p$.
8: Let $\overline{P_i}$ be the path $v_1^i - v_2^i - \ldots - v_{\ell_i}^i$.
9: **end if**
10: **end for**
11: $\overline{P}_p := P_p$.
12: $\overline{\mathcal{P}} := \{\overline{P_1}, \ldots, \overline{P_p}\}$.

Since $\mathcal{P}$ is proper, each path in $\mathcal{P}$ must have at least one non-intersection vertex. Therefore, the if-else clause from line 4 to line 13 in Algorithm 1 is executable. The existence of the path in line 11 of the algorithm is guaranteed by the structure of the transitive closure graph $\overline{G}$. Note that, if for some $i \in \{1, \ldots, p\}$ the path P_i has no intersection vertex, then $\overline{P_i}$ coincides P_i. The following example illustrates the above algorithm.

Example 1.15. Let us again consider the graph G in Figure 1.2. For this graph we have a path covering $\mathcal{P} = \{P_1, P_2, P_3\}$ in which P_1 is the arc $6-7$, P_2

is the path $1-3-5$, and P_3 is the path $3-2-4$. These paths are respectively illustrated by red, blue, and green paths in Figure 1.4. Following Algorithm 1, the path P_1 has no intersection vertex, so we take $\overline{P_1} := P_1$. The path P_2 has vertex 3 belonging to the sequel path P_3, so we will avoid this vertex. From the begin to the end of the path P_2, vertex 1 and vertex 5 respectively appear as the ones that are not in the sequel path P_3. Therefore, line 8 of the algorithm gives us $\overline{P_2}$ consisting of the arc connecting from vertex 1 to vertex 5. Then, line 11 of the algorithm gives us $\overline{P_3} := P_3$. We obtain $\overline{\mathcal{P}} = \{\overline{P_1}, \overline{P_2}, \overline{P_3}\}$ as a vertex-disjoint path covering of the transitive closure graph of G. Note that the path $\overline{P_2}$ consists of arc $(1, 5)$ which is not in the original graph G. The paths in $\overline{\mathcal{P}}$ are illustrated respectively by red, blue, and green paths in Figure 1.5. □

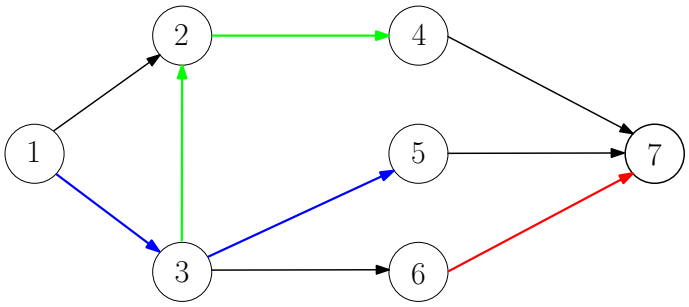

Figure 1.4: A path cover consists of red, blue, green paths for the one in Figure 1.2.

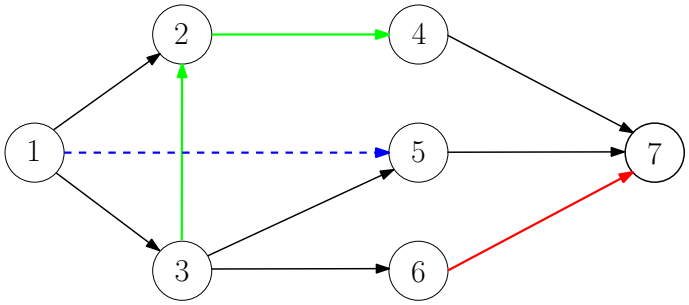

Figure 1.5: The vertex-disjoint path covering obtained by applying Algorithm 1 on the path covering in Figure 1.4.

We now give a detail proof for Lemma 1.14.

Proof of Lemma 1.14. By Algorithm 1, each path P_i in $\mathcal{P}$ is processed once, sequentially in its order in the path covering. If there is no intersection vertex in P_i, then no action is done after scanning this path, i.e., $\overline{P_i} = P_i$. Otherwise, after processing, the path $\overline{P_i}$ only consists of the vertices of P_i that

do not belong to the processed paths, and the intersection vertices of P_i are taken into account for the non-processed paths in $\mathcal{P}$. Thus, after processing all paths in $\mathcal{P}$, we obtain $\overline{P_1}, \ldots, \overline{P_p}$ as vertex-disjoint paths. The collection $\overline{\mathcal{P}} = \{\overline{P_1}, \ldots, \overline{P_p}\}$ hence has the the same size as the path covering $\mathcal{P}$.

On the other hand, all vertices of the paths in $\mathcal{P}$ are scanned, and each vertex belongs to exactly one path $\overline{P_i}$. Since $\mathcal{P}$ is a path covering of G, the vertex set of G is also covered by the paths in $\overline{\mathcal{P}}$. So $\overline{\mathcal{P}}$ is a vertex-disjoint path covering of $\overline{G}$. □

The following lemma shows that the reverse direction of Lemma 1.14 is also true.

Lemma 1.16. *If there is a vertex-disjoint path covering $\overline{\mathcal{P}}$ of $\overline{G}$, then there exists a proper path covering $\mathcal{P}$ of G having the same size as $\overline{\mathcal{P}}$.*

Proof. We construct the path covering $\mathcal{P}$ of G from the path covering $\overline{\mathcal{P}}$ of $\overline{G}$ as follows. Firstly, we assign $\mathcal{P} := \overline{\mathcal{P}}$. Then, for each path P in $\mathcal{P}$, if there exists an arc $(u, v) \in \overline{E} \backslash E$ (i.e., an arc in $\overline{G}$ but not in G), then we replace this arc by a path from u to v in G. This can be done since it adapts the construction of arcs in $\overline{G}$.

Following the above construction, from the path covering $\overline{\mathcal{P}} = \{\overline{P_1}, \ldots, \overline{P_p}\}$ of $\overline{G}$ we obtain a collection $\mathcal{P} = \{P_1, \ldots, P_p\}$ in which each path P_i in $\mathcal{P}$ is obtained by modifying the corresponding path $\overline{P_i}$ in $\overline{\mathcal{P}}$. For each $i = 1, \ldots, p$, the vertex set $V(P_i)$ of the path P_i contains the vertex set $V(\overline{P_i})$ of the path $\overline{P_i}$. Thus we have

$$\bigcup_{i=1}^{p} V(P_i) \supset \bigcup_{i=1}^{p} V(\overline{P_i}) = V. \tag{1.1}$$

The equality in (1.1) is because that $\overline{\mathcal{P}}$ covers the vertex set of G. The first union in (1.1) is the vertex set of all paths in $\mathcal{P}$. Thus $\mathcal{P}$ also covers the vertex set of G. In addition, the arc set in each path P_i of $\mathcal{P}$ are all in E (the arc set of G). Therefore, $\mathcal{P}$ is a path covering of G.

Since the paths $\overline{P_1}, \ldots, \overline{P_p}$ are vertex-disjoint and $V(\overline{P_i}) \subset V(P_i)$ for each $i = 1, \ldots, p$, each path P_i in $\mathcal{P}$ always has at least one vertex that does not belong to another path in $\mathcal{P}$. Hence $\mathcal{P}$ is proper. □

The following theorem is the main result in this subsection.

Theorem 1.17. *The following two problems are equivalent (in the sense that they have the same optimal objective value and there is an one-to-one correspondence between their optimal solutions).*

(P1) *Find a minimum path covering of G.*
(P2) *Find a minimum vertex-disjoint path covering of $\overline{G}$.*

Proof. Let $\mathcal{P}$ be an optimal solution of (P1) and let $p = |\mathcal{P}|$. By applying Algorithm 1 we obtain a vertex-disjoint path covering $\overline{\mathcal{P}}$ of $\overline{G}$ with $|\overline{\mathcal{P}}| = p$. Assume that $\overline{G}$ has another path covering $\overline{\mathcal{P}}'$ with $|\overline{\mathcal{P}}'| < p$. Then, by the construction in the proof of Lemma 1.16, we obtain a proper path covering $\mathcal{P}'$ of G with $|\mathcal{P}'| = |\overline{\mathcal{P}}'| < p$. The existence of $\mathcal{P}'$ contradicts the optimality of $\mathcal{P}$. Thus, $\overline{\mathcal{P}}$ is a minimum vertex-disjoint path covering of $\overline{G}$, i.e., it is an optimal solution to (P2), and p is also the optimal objective value of (P2).

Conversely, let $\overline{\mathcal{Q}}$ be an optimal solution of (P2) and $q = |\overline{\mathcal{Q}}|$. By the construction in the proof of Lemma 1.16, we obtain a proper path covering $\mathcal{Q}$ of G with $|\mathcal{Q}| = q$. Assume that G has another path covering $\mathcal{Q}'$ with $|\mathcal{Q}'| < q$. Then, by applying Algorithm 1, we obtain a path covering $\overline{\mathcal{Q}}'$ of $\overline{G}$ with $|\overline{\mathcal{Q}}'| < q$. This means that $\overline{\mathcal{Q}}$ is not a minimum path covering of $\overline{G}$, contradicting the optimality of $\overline{\mathcal{Q}}$. Hence there is no path covering of G whose size is strictly less than $|\mathcal{Q}|$, so $\mathcal{Q}$ is an optimal solution of (P1) and h is also the optimal value of (P1). □

By Theorem 1.17, instead of considering the problem of finding a minimum path covering of G, we can now focus on the equivalent problem of finding a vertex disjoint path covering of the transitive closure graph $\overline{G}$. In the next subsection, we will formulate the latter problem as a linear program.

1.2.2 Minimum path covering as primal program

We are given the directed acyclic graph $G = (V, E)$ whose vertex set V consists of n vertices named $v_1, \ldots, v_n$, and the transitive closure graph $\overline{G} = (V, \overline{E})$ of G. Our aim is to find a vertex-disjoint path covering of $\overline{G}$ of minimum size. This subsection presents the construction of a linear programming formulation for this problem. This formulation is based an embedding of $\overline{G}$ in a network N constructed as follows.

- The vertex set V_N of N is formed by adding a vertex v_0 to the vertex set V of $\overline{G}$, i.e., $V_N = \{v_0, v_1, \ldots, v_n\}$.
- For each $i = 1, \ldots, n$, we connect the vertex v_0 to the vertex v_i by an arc (v_0, v_i). These arcs are called *forward arcs*.
- For each $i = 1, \ldots, n$, we connect the vertex v_i to the vertex v_0 by an arc (v_i, v_0). These arcs are called *backward arcs*.

- We add a loop (v_0, v_0), i.e. a directed arc starting from v_0 and going back to this vertex.
- For the loop (v_0, v_0) we associate a cost $c_{00} = 1$, while for each of the other arcs in the network we associate a cost 0.

The arc set of the network N is

$$E_N = \overline{E} \cup \{(v_0, v_i), (v_i, v_0) \mid i = 1, \ldots, n\} \cup \{(v_0, v_0)\}.$$

For convenience, we denote E_N^* the set of arcs in the network without the loop (v_0, v_0). Hence, each arc (v_i, v_j) in E_N^* is associated with a cost $c_{ij} = 0$. Figure 1.6 illustrates the network N constructed from the transitive closure graph $\overline{G}$ in Figure 1.3. There, the vertices are represented by their indices, the forward arcs are in red color, and the backward arcs are in blue color.

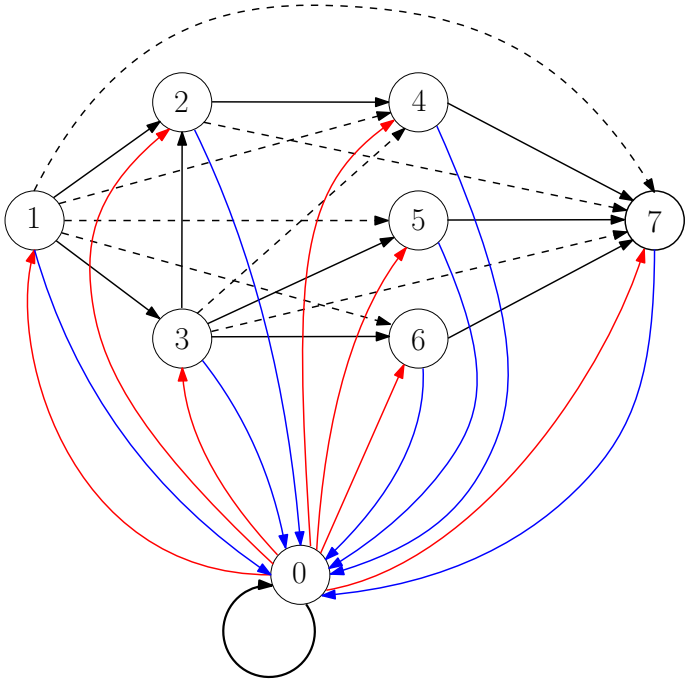

Figure 1.6: The network constructed from the graph in Figure 1.3.

Imagine that the vertex v_0 has a number of flow units and we would like to pump them into the network, then collect them again at v_0. Except for the loop (v_0, v_0), each arc in the network can carry at most one flow unit, but no fraction is allowed. It means that whenever a flow unit comes to the starting vertex of an arc in $\overline{G}$, either the whole flow unit is traversed through the arc or no unit at all. In principle, we can send n flow units in total to $\overline{G}$ through

the n forward arcs. Therefore, we assume that at the beginning the vertex v_0 keeps n flow units.

A flow in the network N is a way of distributing and traversing the flow units in this network. The key idea here is to use the trace of each flow unit traversed in $\overline{G}$ as a path in a vertex-disjoint path covering of this graph. More precisely, assume that from v_0 we pump $p \leq n$ flow units into $\overline{G}$. For each $i = 1, \ldots, p$, the i^{th} flow unit is first sent from v_0 to a vertex $v_1^i \in V$ through the forward arc (v_0, v_1^i). Then, the flow unit traverses along a path $\overline{P_i} = v_1^i - \ldots - v_{\ell_i}^i$ in $\overline{G}$, in which ℓ_i is the number of vertices in this path. From $v_{\ell_i}^i$ the flow unit is collected back to v_0 through the backward arc $(v_{\ell_i}^i, v_0)$. In this way, the flow of the p flow units gives us a collection $\overline{\mathcal{P}} = \{\overline{P_1}, \ldots, \overline{P_p}\}$ of paths in $\overline{G}$. To have $\overline{\mathcal{P}}$ as a vertex-disjoint path covering of $\overline{G}$, each vertex of $\overline{G}$ must be visited by exactly one flow unit. Since p flow units have been sent into $\overline{G}$, there are $n - p$ flow units remaining at the vertex v_0. As a technical action, we pump all of these $n - p$ flow units into the loop (v_0, v_0), so they are back to v_0. For illustration, the path covering in Figure 1.5 corresponds to the flow illustrated by red, blue, green, and purple cycles in Figure 1.7, in which the loop in purple color has 4 flow units and each of the other cycle traverses 1 flow unit.

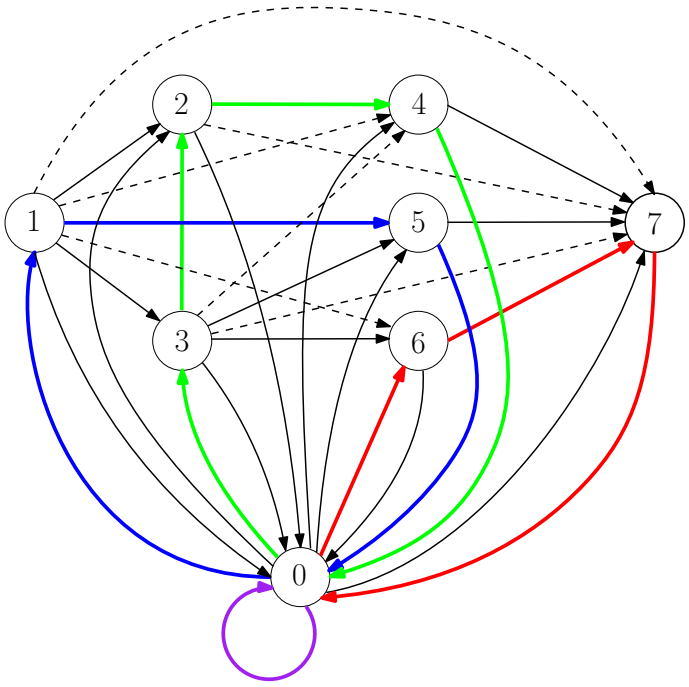

Figure 1.7: The flow corresponds to the path covering in Figure 1.6.

Now, by seeing how many flow units traversed in the loop (v_0, v_0), we can know the size of the path covering $\overline{\mathcal{P}}$. That is the reason why we assign a cost $c_{00} = 1$ to the loop (v_0, v_0) and a cost of 0 to the other arcs in the network. Let us introduce a non-negative integer variable x_{00} to indicate the number of flow units traversed through the loop (v_0, v_0), and for each arc $(v_i, v_j) \in E_N^*$ a binary variable

$$x_{ij} = \begin{cases} 1 & \text{if there is a flow unit traversed through the arc } (v_i, v_j), \\ 0 & \text{otherwise.} \end{cases}$$

Using these variables, the flow value in this network is

$$c_{00}x_{00} + \sum_{(v_i,v_j)\in E_N^*} c_{ij}x_{ij} = x_{00},$$

and therefore, the size of the corresponding path covering $\overline{\mathcal{P}}$ is $n - x_{00}$. Since we aim to find a minimum path covering, our objective is

$$\text{minimize} \quad (n - x_{00}),$$

or equivalently

$$\text{maximize} \quad x_{00}.$$

Since each vertex in $\overline{G}$ is visited by exactly one flow unit, we must have

$$\sum_{\substack{j\in\{0,\ldots,n\}:\\ (v_i,v_j)\in E_N^*}} x_{ij} = 1, \qquad \forall i = 1, \ldots, n, \tag{1.2}$$

$$\sum_{\substack{i\in\{0,\ldots,n\}:\\ (v_i,v_j)\in E_N^*}} x_{ij} = 1, \qquad \forall j = 1, \ldots, n. \tag{1.3}$$

Constraints (1.2) ensure that there is exactly one flow unit going out of each vertex in $\overline{G}$. Constraints (1.3) mean that there is exactly one flow unit coming to each vertex in $\overline{G}$. For the additional vertex v_0 we must impose

$$\sum_{j=0}^{n} x_{0j} = n, \tag{1.4}$$

$$\sum_{i=0}^{n} x_{i0} = n. \tag{1.5}$$

Constraint (1.4) ensures that the vertex v_0 sends n flow units in total to the network, and constraint (1.5) ensures that this vertex collects all of the n flow units. As a summary, we obtain the following integer program for the problem (P2) of finding a minimum vertex-disjoint path covering of $\overline{G}$.

$$\begin{aligned}(MinPC_{IP}) \qquad & \max \quad x_{00} \\ & \text{s.t.} \quad (1.2) - (1.5) \\ & \quad x_{00} \in \mathbb{Z}_+ \\ & \quad x_{ij} \in \{0,1\} \quad \forall i,j \in \{0,\dots,n\} : (v_i,v_j) \in E_N^*.\end{aligned}$$

Consider the following linear relaxation of the above integer program (see [2]).

$$\begin{aligned}(MinPC_{LP}) \qquad & \max \quad x_{00} \\ & \text{s.t.} \quad (1.2) - (1.5) \\ & \quad x_{ij} \geq 0 \quad \forall i,j \in \{0,\dots,n\} : (v_i,v_j) \in E_N.\end{aligned}$$

We have the following observation on this linear program.

Lemma 1.18. *The linear program $(MinPC_{LP})$ is feasible and its optimal objective value is finite.*

Proof. On one hand, let x^* be the vector defined as follows:

$$\begin{aligned} x_{i0}^* &= 1, && \forall i = 1,\dots,n, \\ x_{0j}^* &= 1, && \forall j = 1,\dots,n, \\ x_{00}^* &= 0, && \\ x_{ij}^* &= 0, && \forall i,j = 1,\dots,n : (v_i,v_j) \in E_N^*. \end{aligned}$$

This vector has non-negative components and satisfies (1.2)-(1.5), therefore it is a feasible solution to $(MinPC_{LP})$. So the linear program $(MinPC_{LP})$ is feasible.

On the other hand, it follows from the non-negativity of variables and (1.2) that

$$0 \leq x_{ij} \leq 1, \quad \forall i = 1,\dots,n, j = 0,\dots,n : (v_i,v_j) \in E_N^*.$$

Similarly, from the non-negativity of variables and (1.3) we have

$$0 \leq x_{ij} \leq 1, \quad \forall j = 1,\dots,n, i = 0,\dots,n : (v_i,v_j) \in E_N^*.$$

Lastly, the non-negativity of variables and (1.4)-(1.5) implies

$$0 \leq x_{00} \leq n.$$

So all variables in $(MinPC_{LP})$ is bounded. Since the feasible set of this program is bounded, the maximum in its objective is finite. □

We will show furthermore that $(MinPC_{LP})$ is equivalent to another linear program whose constraints have a better structure to exploit. For the construction of the latter program, let $\widetilde{E} := (V \times V)\backslash E_N$. By this setting, $(v_i, v_j) \in \widetilde{E}$ if there is no directed path from v_i to v_j in the original graph G. In this manner, we call each arc in $\widetilde{E}$ an *fictitious arc*. We would like to take these fictitious arcs into account of the new program. However, since they are fictitious, we must impose that no flow unit is transferred through any arc in $\widetilde{E}$. Following these ideas, for each fictitious arc $(v_i, v_j) \in \widetilde{E}$ we introduce a non-negative variable x_{ij} and assign a cost $c_{ij} = -\infty$. We come up with the following linear program (see [2]).

$$(PathCover_{LP}) \qquad \max \quad \sum_{i,j=0}^{n} c_{ij}x_{ij} \tag{1.6}$$

$$\text{s.t.} \quad \sum_{j=0}^{n} x_{ij} = 1, \qquad \forall i = 1, \ldots, n, \tag{1.7}$$

$$\sum_{i=0}^{n} x_{ij} = 1, \qquad \forall j = 1, \ldots, n, \tag{1.8}$$

$$\sum_{j=0}^{n} x_{0j} = n, \tag{1.9}$$

$$\sum_{i=0}^{n} x_{i0} = n, \tag{1.10}$$

$$x_{ij} \geq 0, \qquad \forall i, j = 0, \ldots, n. \tag{1.11}$$

Since the costs associated to the arcs of our network are given by

$$c_{ij} = \begin{cases} 1 & \text{if } i = j = 0, \\ 0 & \text{if } (v_i, v_j) \in E_N^*, \\ -\infty & \text{if } (v_i, v_j) \in \widetilde{E}, \end{cases}$$

the objective function in (1.6) is equal to

$$x_{00} + \sum_{(v_i,v_j)\in\widetilde{E}} c_{ij}x_{ij}.$$

Since we aim to maximize this objective function on non-negative variables, the use of $c_{ij} = -\infty$ for fictitious arcs $(v_i, v_j) \in \widetilde{E}$ is just an alternative way to impose that the corresponding variables must be 0. This is the key idea to prove the equivalence between the two linear programs $(MinPC_{LP})$ and $(PathCover_{LP})$. To be precise, we first start with the following observation.

Lemma 1.19. (see [2]) *The optimal objective value of* $(PathCover_{LP})$ *does not exceed the one of* $(MinPC_{LP})$.

Proof. We have already known from Lemma 1.18 that the optimal objective value of $(MinPC_{LP})$ exists finitely. In case that $(PathCover_{LP})$ is infeasible, its optimal objective value is $-\infty$, which is clearly less than the one of $(MinPC_{LP})$. It is left to consider the case that $(PathCover_{LP})$ is feasible.

Let $\overline{x}$ be an optimal solution of $(PathCover_{LP})$. If $\overline{x}_{ij} > 0$ for some indices i, j such that $(v_i, v_j) \in \widetilde{E}$, then for that pair of indices we have $c_{ij} = -\infty$ and therefore $c_{ij}\overline{x}_{ij} = -\infty$. Consequently, the objective value of $(PathCover_{LP})$ at $\overline{x}$ is $-\infty$, which is again clearly less than the one of $(MinPC_{LP})$. So now we only need to focus on the case in which $\overline{x}_{ij} = 0$ for all i, j such that $(v_i, v_j) \in \widetilde{E}$.

Let $x^{E_N} \in \mathbb{R}^{|E_N|}$ be defined by setting $x_{ij}^{E_N} = \overline{x}_{ij}$ for each $(v_i, v_j) \in E_N$, then we have immediately that all components of x^{E_N} are non-negative. Since $\overline{x}$ satisfies (1.7), for each $i = 1, \ldots, n$ we have

$$1 = \sum_{j=0}^{n} \overline{x}_{ij} = \sum_{\substack{j \in \{0,\ldots,n\}: \\ (v_i,v_j) \in E_N^*}} \overline{x}_{ij} + \sum_{\substack{j \in \{0,\ldots,n\}: \\ (v_i,v_j) \in \widetilde{E}}} \overline{x}_{ij} = \sum_{\substack{j \in \{0,\ldots,n\}: \\ (v_i,v_j) \in E_N^*}} \overline{x}_{ij} = \sum_{\substack{j \in \{0,\ldots,n\}: \\ (v_i,v_j) \in E_N^*}} x_{ij}^{E_N},$$

which means that x^{E_N} satisfies (1.2). Similarly, from the fact that $\overline{x}$ satisfies (1.8), we deduce that x^{E_N} satisfies (1.3). Furthermore, for each $i, j = 0, \ldots, n$ we have $(v_0, v_j) \notin E$ and $(v_i, v_0) \notin E$, hence it follows that

$$\begin{aligned} x_{0j}^{E_N} &= \overline{x}_{0j}, && \forall j = 0, \ldots, n, \\ x_{i0}^{E_N} &= \overline{x}_{i0}, && \forall i = 0, \ldots, n. \end{aligned}$$

So by (1.9) we have

$$1 = \sum_{j=0}^{n} \overline{x}_{0j} = \sum_{j=0}^{n} x_{0j}^{E_N},$$

which means that x^{E_N} satisfies (1.4). Similarly, from the fact that $\overline{x}$ satisfies (1.10), we deduce that x^{E_N} satisfies (1.5). To summarize, x^{E_N} has

non-negative components and satisfies (1.2)-(1.5), therefore x^{E_N} is a feasible solution to $(MinPC_{LP})$.

Since $\overline{x}$ is an optimal solution of $(PathCover_{LP})$, the optimal objective value of this program is

$$\overline{x}_{00} + \sum_{(v_i,v_j)\in\widetilde{E}} c_{ij}\overline{x}_{ij} = \overline{x}_{00} = x_{00}^{E_N},$$

which is the objective value of $(MinPC_{LP})$ at x^{E_N}. As we have shown above, x^{E_N} is a feasible solution to $(MinPC_{LP})$, so the objective value of $(MinPC_{LP})$ at x^{E_N} does not exceed the optimal objective value of this program. Therefore, the optimal objective value of $(PathCover_{LP})$ is at most the one of $(MinPC_{LP})$. □

The reverse direction of Lemma 1.19 also holds true, as shown in the next lemma.

Lemma 1.20. (see [2]) *The linear programs $(MinPC_{LP})$ and $(PathCover_{LP})$ are equivalent (in the sense that they have the same optimal objective value and there is an one-to-one correspondence between their optimal solutions).*

Proof. Let x^{E_N} be an optimal solution of $(MinPC_{LP})$, then all components of x^{E_N} are non-negative. By Lemma 1.18 the existence of x^{E_N} is guaranteed. Let $\overline{x}$ be defined as follows.

$$\overline{x}_{ij} = \begin{cases} x_{ij}^{E_N} & \text{if } (v_i,v_j)\in E_N, \\ 0 & \text{if } (v_i,v_j)\in\widetilde{E}. \end{cases}$$

Since all components of x^{E_N} are non-negative, so are components of $\overline{x}$. By construction of our network, the arcs (v_0,v_j) and (v_i,v_0) (for $i,j=0,\ldots,n$) are non-fictitious, i.e., they are not in $\widetilde{E}$. Thus we have

$$\overline{x}_{0j} = x_{0j}^{E_N}, \qquad \forall j=0,\ldots,n, \tag{1.12}$$

$$\overline{x}_{i0} = x_{i0}^{E_N}, \qquad \forall i=0,\ldots,n. \tag{1.13}$$

Since x^{E_N} is an optimal solution of $(MinPC_{LP})$, it satisfies constraints (1.4) and (1.5). So we have

$$\sum_{j=0}^{n} x_{0j}^{E_N} = n \tag{1.14}$$

$$\sum_{i=0}^{n} x_{i0}^{E_N} = n. \tag{1.15}$$

By (1.12)-(1.15) we obtain

$$\sum_{j=0}^{n} \overline{x}_{0j} = n,$$
$$\sum_{i=0}^{n} \overline{x}_{i0} = n.$$

In other words, $\overline{x}$ satisfies (1.9)-(1.10). Additionally, for each $i = 1, \ldots, n$ we have

$$\sum_{j=0}^{n} \overline{x}_{ij} = \sum_{\substack{j\in\{0,\ldots,n\}:\\ (v_i,v_j)\in E_N^*}} \overline{x}_{ij} + \sum_{\substack{j\in\{0,\ldots,n\}:\\ (v_i,v_j)\in \widetilde{E}}} \overline{x}_{ij} = \sum_{\substack{j\in\{0,\ldots,n\}:\\ (v_i,v_j)\in E_N^*}} \overline{x}_{ij} = \sum_{\substack{j\in\{0,\ldots,n\}:\\ (v_i,v_j)\in E_N^*}} x_{ij}^{E_N} = 1,$$

which means that $\overline{x}$ satisfies (1.7). The second and the third equalities above are due to the definition of $\overline{x}$, while the last equality is due to the fact that x^{E_N} satisfies (1.2) as it is an optimal solution of $(MinPC_{LP})$. Similarly, we also obtain that $\overline{x}$ satisfies (1.8).

We have shown that $\overline{x}$ has non-negative components and satisfies (1.7)-(1.10). Thus $\overline{x}$ is a feasible solution to $(PathCover_{LP})$. Note that the objective value of $(PathCover_{LP})$ at $\overline{x}$ is

$$\overline{x}_{00} + \sum_{(v_i,v_j)\in\widetilde{E}} c_{ij}\overline{x}_{ij} = \overline{x}_{00} = x_{00}^{E_N},$$

which is the same as the optimal objective value of $(MinPC_{LP})$. This, together with the fact that the optimal objective value of $(PathCover_{LP})$ does not exceed the one of $(MinPC_{LP})$ as proved in Lemma 1.19, implies that $\overline{x}$ is in fact an optimal solution of $(PathCover_{LP})$. Hence, the two linear programs have the same optimal objective value, and the correspondence between x^{E_N} and $\overline{x}$ completes the proof of this lemma. □

Thanks to Lemma 1.20, the following proposition gives the key result in this subsection.

Proposition 1.21. (see [2]) *The linear program $(MinPC_{LP})$ admits an integral optimal solution, i.e. an optimal solution whose all components are integer.*

Proof. In form of (1.6)-(1.11), $(PathCover_{LP})$ is a transportation problem (cf. [7], Section I.3.5). The right hand sides of constraints (1.7)-(1.10) are

integers, therefore this program has an integral optimal solution (see e.g. [7], Corollary 5.2). Let $\overline{x}$ be such a solution of $(PathCover_{LP})$. Following the proofs of Lemma 1.19 and Lemma 1.20, the vector $x^{E_N} \in \mathbb{R}^{|E_N|}$ defined by

$$x_{ij}^{E_N} = \overline{x}_{ij} \quad \forall (v_i, v_j) \in E_N$$

is an optimal solution of $(MinPC_{LP})$. Since the components of x^{E_N} are also components of $\overline{x}$, they are all integers. Thus, $(MinPC_{LP})$ admits x^{E_N} as an integral optimal solution. □

By Proposition 1.21, instead of solving the integer program $(MinPC_{IP})$ we just need to solve to optimality the linear relaxation $(MinPC_{LP})$. Any integral solution to $(MinPC_{LP})$ is a feasible solution to $(MinPC_{IP})$, which encodes a vertex-disjoint path covering of $\overline{G}$. Hence, any integral optimal solution to $(MinPC_{LP})$ encodes a minimum vertex-disjoint path covering of $\overline{G}$.

1.2.3 Maximum antichain as dual program

It has been shown from the previous subsection that the linear program $(MinPC_{LP})$, which reads as

$$\begin{aligned}
(MinPC_{LP}) \qquad & \max \quad x_{00} \\
\text{s.t.} \quad & \sum_{\substack{j \in \{0,\dots,n\}: \\ (v_i,v_j) \in E_N^*}} x_{ij} = 1, \qquad \forall i = 1, \dots, n && (1.2) \\
& \sum_{\substack{i \in \{0,\dots,n\}: \\ (v_i,v_j) \in E_N^*}} x_{ij} = 1, \qquad \forall j = 1, \dots, n, && (1.3) \\
& \sum_{j=0}^{n} x_{0j} = n, && (1.4) \\
& \sum_{i=0}^{n} x_{i0} = n, && (1.5) \\
& x_{ij} \geq 0, \qquad \forall i, j \in \{0, \dots, n\} : (v_i, v_j) \in E_N,
\end{aligned}$$

admits an integral optimal solution which encodes a minimum vertex-disjoint path covering of $\overline{G}$. The optimal value of this program is $n - p$ in which p is the size of the optimal path covering. The dual to this program (see [2]) is

$$(DualPC_{LP}) \qquad \min \qquad \sum_{i=1}^{n} u_i + \sum_{j=1}^{n} w_j + n(u_0 + w_0), \qquad (1.16)$$

$$\begin{aligned}
\text{s.t.} \quad & u_0 + w_0 \geq 1, && && (1.17)\\
& u_i + w_0 \geq 0, && \forall i = 1, \ldots, n, && (1.18)\\
& u_0 + w_j \geq 0, && \forall j = 1, \ldots, n, && (1.19)\\
& u_i + w_j \geq 0, && \forall i, j > 0 : (v_i, v_j) \in E_N^*, && (1.20)\\
& u_i, w_j \in \mathbb{R}, && \forall i, j = 0, \ldots, n. && (1.21)
\end{aligned}$$

In this dual program, u_0 and w_0 are respectively the dual variables corresponding to constraints (1.4) and (1.5), while u_i and w_j (with $i, j \in \{1, \ldots, n\}$) are respectively the dual variables corresponding to constraints (1.2) and (1.3).

The first observation on $(DualPC_{LP})$ is given in the following proposition.

Proposition 1.22. (see [2]) *The optimal objective value of* $(DualPC_{LP})$ *equals the one of* $(MinPC_{LP})$.

Proof. As shown in the previous subsection, the linear program $(MinPC_{LP})$ is feasible and has finite optimal objective value. Thus, the claim in this proposition follows from the well-known strong duality theorem in linear programming. □

As proved in Proposition 1.21, the primal program $(MinPC_{LP})$ admits an integral optimal objective value. The following proposition gives a similar result for the dual program $(DualPC_{LP})$.

Proposition 1.23. (see [2]) *The dual program* $(DualPC_{LP})$ *admits an integral optimal solution.*

Proof. In order to obtain a contradiction, assume that all optimal solutions to $(DualPC_{LP})$ are non-integral. Among these optimal solutions, let us consider one with the fewest number of non-integers among the values of $u_0, \ldots, u_n$ and $w_0, \ldots, w_n$. Since we aim to minimize the objective function (1.16) and the right hand sides of constraints (1.17)-(1.20) are integers, there must be at least one non-integer in $u_0, \ldots, u_n$ and at least one non-integer in $w_0, \ldots, w_n$. Let

$$\varepsilon = \min\{u_i - \lfloor u_i \rfloor \mid i = 0, \ldots, n \text{ such that } u_i \text{ is not integer}\}.$$

Here, $\lfloor u_i \rfloor$ is the largest integer that is smaller than or equal to u_i. Let us define a new solution $(u'_0, \ldots, u'_n, w'_0, \ldots, w'_n)$ to $(DualPC_{LP})$ as follows:

$$u'_i = \begin{cases} u_i - \varepsilon & \text{if } u_i \text{ is non-integer,} \\ u_i & \text{if } u_i \text{ is integer,} \end{cases}$$

$$w'_j = \begin{cases} w_j + \varepsilon & \text{if } w_j \text{ is non-integer,} \\ w_j & \text{if } v_j \text{ is integer.} \end{cases}$$

If $u'_i = u_i - \varepsilon$ and $w'_j = w_j + \varepsilon$ participate in a constraint among (1.17)-(1.20) for some $i, j \in \{0, \ldots, n\}$, then we have $u'_i + w'_j = u_i + w_j$ and hence they satisfy that constraint. If $u'_i = u_i$ and $w'_j = w_j + \varepsilon$ participate in a constraint among (1.17)-(1.20) for some $i, j \in \{0, \ldots, n\}$, then we have $u'_i + w'_j = u_i + w_j + \varepsilon > u_i + w_j$, and hence they also satisfy that constraint.

It is left to check for $i, j \in \{0, \ldots, n\}$ such that $u'_i = u_i - \varepsilon$ and $w'_j = w_j$. In such case, we are given non-integer u_i and integer w_j with $u_i + w_j \geq c_{ij}$ for $c_{ij} \in \mathbb{Z}$. More precisely, in the setting of $(DualPC_{LP})$ we have $c_{ij} = 0$ except for $c_{00} = 1$. It follows that $\lfloor u_i \rfloor + w_j \geq c_{ij}$. Now, by definition of ε we have

$$u'_i + w'_j = u_i - \varepsilon + w_j \geq u_i - (u_i - \lfloor u_i \rfloor) + w_j = \lfloor u_i \rfloor + w_j \geq c_{ij}.$$

Hence, in this case u'_i and w'_j also satisfy the constraints of $(DualPC_{LP})$.

The above arguments show that $(u'_0, \ldots, u'_n, w'_0, \ldots, w'_n)$ is a feasible solution to $(DualPC_{LP})$. Together with our assumption that $(u_0, \ldots, u_n, w_0, \ldots, w_n)$ is an optimal solution of $(DualPC_{LP})$, this fact gives us

$$\sum_{i=1}^{n} u'_i + \sum_{j=1}^{n} w'_j + n(u'_0 + w'_0) \geq \sum_{i=1}^{n} u_i + \sum_{j=1}^{n} w_j + n(u_0 + w_0). \tag{1.22}$$

Note that, by the choice of ε, the former solution has at least one more integral (u-)component, or equivalently, at least one less non-integral component, than the latter solution. Therefore, if these two solutions have the same objective value, then the existence of the former solution contradicts our assumption that the latter one is the solution with the fewest number of non-integers. So these two solutions must have different objective values. Keeping (1.22) in mind, it follows that

$$\sum_{i=1}^{n} u'_i + \sum_{j=1}^{n} w'_j + n(u'_0 + w'_0) > \sum_{i=1}^{n} u_i + \sum_{j=1}^{n} w_j + n(u_0 + w_0).$$

By definition of variables u' and v', this inequality implies

$$\begin{aligned} &|\{u_i \in \{u_1, \ldots, u_n\} : u_i \text{ is non-integer}\}| + n\delta_{u_0} \\ &< |\{w_j \in \{w_1, \ldots, w_n\} : v_j \text{ is non-integer}\}| + n\delta_{w_0}, \end{aligned} \tag{1.23}$$

in which

$$\delta_{u_0} = \begin{cases} 1 & \text{if } u_0 \text{ is non-integer,} \\ 0 & \text{if } u_0 \text{ is integer,} \end{cases}$$

$$\delta_{w_0} = \begin{cases} 1 & \text{if } w_0 \text{ is non-integer,} \\ 0 & \text{if } w_0 \text{ is integer.} \end{cases}$$

Consider another solution $(\overline{u}_0, \ldots, \overline{u}_n, \overline{w}_0, \ldots, \overline{w}_n)$ defined as follows:

$$\overline{u}_i = \begin{cases} u_i + \delta & \text{if } u_i \text{ is non-integer,} \\ u_i & \text{if } u_i \text{ is integer,} \end{cases}$$

$$\overline{w}_j = \begin{cases} w_j - \delta & \text{if } w_j \text{ is non-integer,} \\ w_j & \text{if } w_j \text{ is integer,} \end{cases}$$

in which

$$\delta = \min\{w_j - \lfloor w_j \rfloor \mid j = 0, \ldots, n \text{ such that } w_j \text{ is not integer}\}.$$

By the same arguments as applied for u' and w' above, $(\overline{u}_0, \ldots, \overline{u}_n, \overline{w}_0, \ldots, \overline{w}_n)$ is also a feasible solution to $(DualPC_{LP})$. However, by construction of $\overline{u}$ and $\overline{w}$, it follows from (1.23) that

$$\sum_{i=1}^{n} \overline{u}_i + \sum_{j=1}^{n} \overline{w}_j + n(\overline{u}_0 + \overline{w}_0) < \sum_{i=1}^{n} u_i + \sum_{j=1}^{n} w_j + n(u_0 + w_0).$$

This inequality contradicts the optimality of $(u_0, \ldots, u_n, w_0, \ldots, w_n)$. This contradiction implies that all optimal solution to $(DualPC_{LP})$ are integral. □

We now show a specific integral optimal solution to $(DualPC_{LP})$. For that, we start with an arbitrary integral optimal solution $(u_0, \ldots, u_n, w_0, \ldots, w_n)$ of this program. Following the above proof of feasibility of $(u'_0, \ldots, u'_n, w'_0, \ldots, w'_n)$, we can add a constant to each of $u_0, \ldots, u_n$ and then subtract the same constant from each of $w_0, \ldots, w_n$ without violating (1.17)-(1.20) or changing the objective value (1.16). By choosing w_0 for such constant, we can assume that $w_0 = 0$. Then, constraint (1.17) implies that $u_0 \geq 1$. If $u_0 > 1$, we can subtract an amount of $\varepsilon = u_0 - 1$ from u_0 and then add the same amount ε to each of $w_1, \ldots, w_n$. This preserves (1.17)-(1.20) and also preserves the objective value (1.16). So we can assume $u_0 = 1$. Then, constraints (1.19) implies that

$$w_j \geq -1 \qquad \forall j = 1, \ldots, n. \tag{1.24}$$

Since $w_0 = 0$, constraints (1.18) implies that

$$u_i \geq 0 \qquad \forall i = 1, \ldots, n. \tag{1.25}$$

Since we aim to minimize the objective (1.16), by (1.25) and (1.20) we can impose

$$w_j \leq 0 \qquad \forall j = 1, \ldots, n. \tag{1.26}$$

Similarly, by (1.24), (1.20), and the desire to minimize (1.16), we can impose

$$u_i \leq 1 \qquad \forall i = 1, \ldots, n. \tag{1.27}$$

By (1.25), (1.27), and the integrality of u-components, we can set value 0 to some elements in $u_1, \ldots, u_n$ and set value 1 to the others. Similarly, by (1.24), (1.26), and the integrality of w-components, we can set value 0 to some elements in $w_1, \ldots, w_n$ and set value -1 to the others.

Let us define

$$A = \{v_j \in V \mid w_j = -1\}$$

and

$$A^* = \{v_i \in V \mid \exists\ v_j \in A : (v_i, v_j) \in E_N^*\}.$$

In words, A is the set of vertices in the transitive closure graph $\overline{G}$ whose corresponding w-variables obtain value -1, and A^* is the set of vertices in $\overline{G}$ that are reachable to a vertex in A. It follows from the definition of A and A^* that $A \backslash (A \cap A^*)$ is an antichain in G. Therefore, if we let q be the size of a maximum antichain in G, then we have

$$|A \backslash (A \cap A^*)| \leq q. \tag{1.28}$$

By (1.20) and our desire of minimizing (1.16), A^* is exactly the set of vertices in $\overline{G}$ whose corresponding u-variables equals 1, i.e.,

$$A^* = \{v_i \in V \mid u_i = 1\}.$$

Hence, the objective value (1.16) at the solution we are considering is

$$\sum_{i=1}^{n} u_i + \sum_{j=1}^{n} w_j + n(u_0 + w_0) = \sum_{v_i \in A^*} u_i + \sum_{v_j \in A} w_j + n(1 + 0) = |A^*| - |A| + n.$$

By Proposition 1.22 and keeping in mind that the solution we are considering is an optimal solution to ($DualPC_{LP}$), it follows that

$$|A^*| - |A| + n = n - p \quad \Leftrightarrow \quad p = |A| - |A^*|,$$

in which p is the size of a minimum vertex-disjoint path covering of $\overline{G}$. Let q be the size of a maximum antichain in G. Then we have

$$p = |A| - |A^*| \leq |A| - |A \cap A^*| \leq |A \backslash (A \cap A^*)| \leq q.$$

The last inequality above is due to (1.28). To prove Theorem 1.10, i.e., to prove $p = q$, it is left to show the reverse inequality $p \geq q$.

It is obvious to see that $p \geq q$. Indeed, let B be a maximum antichain in G, then we have $B \subseteq V$ and $|B| = q$. Let $\mathcal{P}$ be an arbitrary path covering of G. Since $B \subseteq V$, the path covering $\mathcal{P}$ also covers B. If $|\mathcal{P}| < q = |b|$, then by Dirichlet principle there must exist two distinct vertices $u, v \in B$ belonging to a same path in $\mathcal{P}$, and consequently, either u is reachable from v or v is reachable from u. This contradicts the fact that B is an antichain. This contradiction implies that $|\mathcal{P}| \geq q$. Since this inequality holds for arbitrary path covering $\mathcal{P}$ of V, it also holds when $\mathcal{P}$ is a minimum path covering of V. In that case we have $|\mathcal{P}| = p$, hence $p \geq q$ as desired.

1.3 Second proof

Perles in [3] gives an induction proof for Dilworth's theorem. Originally, his proof is for the poset version of the theorem. Following the proof of Perles, in this section we present in detail a proof for the graph version of Dilworth's theorem.

Let G be a directed acyclic graph and V the vertex set of G. Let p be the size of a minimum path covering of V and q the size of a maximum antichain in G. The graph version of Dilworth's theorem (Theorem 1.10) states that $p = q$, which is what we need to prove in this section. By the same arguments as in the last paragraph of the previous subsection, we have $p \geq q$. It is left to show that $p \leq q$.

In the following we will prove $p \leq q$ by induction on $|V|$. This inequality is obvious in case $|V| = 1$, since in this case G has only one vertex and therefore $p = q = 1$. Let $k > 1$ be an integer. We make the induction assumption that the inequality $p \leq q$ holds for all directed acyclic graphs having at most $k - 1$ vertices. Consider the case that G has k vertices. For any maximum antichain A in G, let

$$A^{\leftarrow} := \{u \in V \mid u \text{ is reachable to a vertex in } A\} \subset V,$$
$$A^{\rightarrow} := \{u \in V \mid u \text{ is reachable from a vertex in } A\} \subset V.$$

We have the following two claims.

Claim 1: $A^{\leftarrow} \cup A^{\rightarrow} = V$. Indeed, assume the contrary that there exists a vertex $u^* \in V \backslash (A^{\leftarrow} \cup A^{\rightarrow})$. Since $u^* \notin A^{\leftarrow}$, from u^* we cannot reach to any vertex in A. On the other hand, since $u^* \notin A^{\rightarrow}$, there is no vertex in A that

can reach to u^*. Consequently, $A^* = A \cup \{u^*\}$ is also an antichain in G, which has size $|A^*| = |A| + 1 > |A|$. This contradicts the assumption that A is a maximum antichain in G. This contradiction implies that $A^{\leftarrow} \cup A^{\rightarrow} = V$.

Claim 2: $A^{\leftarrow} \cap A^{\rightarrow} = A$. Indeed, since A is an antichain, each vertex in A is only reachable to and from itself, i.e., $A \subset A^{\leftarrow}$ and $A \subset A^{\rightarrow}$. Consequently, we have $A \subset A^{\leftarrow} \cap A^{\rightarrow}$. To show the inverse inclusion $A^{\leftarrow} \cap A^{\rightarrow} \subset A$, let u be an arbitrary vertex in $A^{\leftarrow} \cap A^{\rightarrow}$. Since $u \in A^{\leftarrow}$, from u we can reach to some vertex v in A by some u-v-path. Since $u \in A^{\rightarrow}$, we can reach to u from some vertex w in A by some w-u-path. Concatenating these two paths at u, we obtain a path from $w \in A$ to $v \in A$. Since A is an antichain and G is acyclic, this happens only when $u = v = w$, and in this case we have $u \in A$ due to the fact that both v and w are in A. Since u is chosen arbitrarily in $A^{\leftarrow} \cap A^{\rightarrow}$, we deduce that $A^{\leftarrow} \cap A^{\rightarrow} \subset A$.

For the sets $A^{\leftarrow}$ and $A^{\rightarrow}$ constructed above, only two following cases can happen (note that these two cases are mutually exclusive).

- **Case 1:** there exists a maximum antichain A in G such that $A^{\leftarrow} \subsetneq V$ and $A^{\rightarrow} \subsetneq V$.
- **Case 2:** for any maximum antichain A in G we have either $A^{\leftarrow} = V$ or $A^{\rightarrow} = V$.

Let us show $p \leq q$ in Case 1. In this case we have $|A^{\leftarrow}| < |V| = k$ since $A^{\leftarrow} \subsetneq V$. According to our induction assumption, we can cover $A^{\leftarrow}$ by a path covering $\mathcal{P}_{A^{\leftarrow}}$ consisting of q paths. Furthermore, since $A^{\leftarrow} \cap A^{\rightarrow} = A$, we have $A \subset A^{\leftarrow}$, hence these q paths also cover A. Since A is an antichain with $|A| = q$, each vertex $u \in A$ must belong to exactly one path P_u in the path covering $\mathcal{P}_{A^{\leftarrow}}$. By similar arguments for $A^{\rightarrow}$, we can cover $A^{\rightarrow}$ by a path covering $\mathcal{P}_{A^{\rightarrow}}$ consisting of q paths, in which each vertex $u \in A$ must belong to exactly one path P'_u of this path covering. Then, for each $u \in A$, $P_u \cup P'_u$ forms a path in G going through u. Again, since A is an antichain, the set $\mathcal{P}_A = \{P_u \cup P'_u \mid u \in A\}$ consists of $|A| = q$ paths. Since $\mathcal{P}_{A^{\leftarrow}} = \{P_u \mid u \in A\}$ covers $A^{\leftarrow}$, $\mathcal{P}_{A^{\rightarrow}} = \{P'_u \mid u \in A\}$ covers $A^{\rightarrow}$, and additionally $A^{\leftarrow} \cup A^{\rightarrow} = V$, the set $\mathcal{P}_A$ constructed above is a path covering of V with $|\mathcal{P}_A| = q$. Let $\mathcal{P}$ be a minimum path covering of V. Due to the minimum size of $\mathcal{P}$, we have $p = |\mathcal{P}| \leq |\mathcal{P}_A| = q$.

It is left to show $p \leq q$ in Case 2. In this case, for any antichain A of size q (i.e., a maximum antichain in G), we have either $A^{\leftarrow} = V$ or $A^{\rightarrow} = V$. We say that a vertex $u \in V$ is a *minimal vertex* if there is no vertex in G reachable

to u. Similarly, we say that a vertex $u \in V$ is a *maximal vertex* if there is no vertex in G reachable from u. Note that $A^{\leftarrow} \cap A^{\rightarrow} = A$, so if $A^{\leftarrow} = V$, then $A^{\rightarrow} = A$ (i.e. from each vertex in A we cannot reach other vertices in G, or in other words the antichain A is the set of all maximal vertices in G). Similarly, if $A^{\rightarrow} = V$, then $A^{\leftarrow} = A$ (i.e. we cannot reach to any vertex in A from other vertices in G, or in other words the antichain A is the set of all minimum vertices in G). For a recap, each antichain A of size q is either the set of all maximal vertices or the set of all minimal vertices in G. Thus, if we choose u as a minimal vertex in G, then from u we can reach to a maximal vertex v in G (since G is acyclic), and furthermore each antichain of size q in G must contain either u or v. As a consequence, each maximum antichain in the graph $G[V\backslash\{u, v\}]$ induced by the vertex set $V\backslash\{u, v\}$ has size $q - 1$. Let $\mathcal{P}^*$ be a minimum path covering of $V\backslash\{u, v\}$. By our induction assumption, we have $|\mathcal{P}^*| \leq q - 1$. By adding the u-v-path mentioned above to the path covering $\mathcal{P}^*$, we obtain a path covering $\mathcal{P}$ that covers V with $|\mathcal{P}| \leq q$. Recall that p is the size of a minimum path covering of V, and $\mathcal{P}$ constructed above is a (not necessarily minimum) path covering of V, so $p \leq |\mathcal{P}|$. Therefore we have $p \leq |\mathcal{P}| \leq q$, which closes the proof.

Chapter 2

An application in cob-robber guarding game

This chapter presents an application of the maximum antichain problem in studying a cop-robber guarding game. In Section 2.1 we describe the game and state the problem under our consideration. In Section 2.2 we present a detail construction to model this problem as a path covering problem. In Section 2.3, by apply Dilworth's theorem, a lower bound on the optimal value of our considered problem is given. The contents of this chapter are written on the basis of the paper [5].

2.1 Problem statement

We first present a detail description of the cop-robber guarding game studied in this chapter. The game is played on a simple undirected graph G whose vertex set is named by V and edge set is named by E. The vertex set V is finite and partitioned into two disjoint non-empty subsets $C = \{c_0, \ldots, c_{|C|-1}\}$ and $R = V \backslash C = \{r_0, \ldots, r_{|R|-1}\}$. The *protected region* is the subgraph $G[C]$ of G induced by the vertex set C. The subgraph $G[R]$ of G induced by the vertex set R is called the *robber region*. We consider the case that **the robber region is a cycle**. It means that, throughout this chapter, the robber region $G[R]$ is the cycle $r_0 - r_1 - \ldots - r_{|R|-1} - r_0$.

The game is played by two players: the *cop player* and the *robber player*. The latter player has a unique pawn called the *robber pawn*, or robber for short. The former player has n_c pawns called the *cop pawns*, or cops for short. Here, n_c is a positive integer.

The two players play in alternating turns. For the ease of our description,

the turns are counted from 0. In the first turn (i.e., turn 0), the cop player places the robber on a vertex of the robber region $G[R]$. In the next turn, the cop player places his n_c cops on vertices of the protected region $G[C]$. On a vertex in the protected region, the cop player can place more than one cop. By this initial setting, the robber player plays in the even turns, while the cop player plays in the odd turns. By a round we mean a turn of the robber player and the subsequent turn of the cop player. More precisely, round $i \in \mathbb{N}$ consists of turn $2i$ (of the robber player) and turn $2i + 1$ (of the cop player).

In each even turn, the robber player can either keep the robber at its current position or move it to a neighboring vertex. In each odd turn, the cop player can determine the movement of the n_c cops, in which each cop can either stay at its current vertex or move to a neighboring vertex *within the protected region*. It means that no cop is allowed to move to a vertex in the robber region. Therefore, we sometime call the protected region by *cop region*.

At any turn, both players know the positions of the cops and the robber in the graph. It at some turn of the robber player, the robber can move to a vertex in the cop region on which there is no cop, then we say that the robber successfully attacks the protected region and the robber player wins the game. If the robber moves to a vertex in the protected region on which at least one cop is currently staying, then we say that the robber is caught. In this sense, whenever a cop stays on a vertex in the protected region, we say that the cop guards the vertex. The cop player wins the game if he can prevent the protected region from the attack of the robber, i.e., if he can make sure that the robber player cannot win the game.

A strategy of the robber player is understood as a way that he moves the robber in his turns. Similarly, a strategy of the cop player is understood as a way that he moves the cops in his turns. We are now ready to state shortly our interested problem in the cop-robber guarding game: *Find a strategy for the cop player with minimum number of used cops so that he wins the game.*

Before going to the next section, we have a remark that helps to simplify our consideration. The protected region $G[C]$ can consist of different connected components. If there is such a component that does not have any edge connected to the robber region, then clearly the robber cannot come to any vertex in that component. Consequently, the cop player does not need to place any cop to guard such component. In such situation, we can omit the connected component without affecting the optimal number of cops as well

as the winning strategy of the cop player. Therefore, we can assume that **each connected component of the protected region has at least one vertex adjacent to the robber region**.

Example 2.1. Consider the cop-robber guarding game on the graph G in Figure 2.1. This graph has 14 vertices that are partitioned into two subsets: $R = \{r_0, r_1, r_2, r_3\}$ and $C = \{c_0, c_1, c_2, c_3, c_4, c_5, c_6, c_7, c_8, c_9\}$. The edges of G are represented as in the figure. The protected region is the subgraph $G[C]$ illustrated by blue part in the figure. It has 2 connected components, in which the first component consists of 8 vertices $c_0, \ldots, c_7$ and the second component consists of 2 vertices c_8, c_9. The robber region is the subgraph $G[R]$ illustrated by red part in the figure. It is a cycle going through 4 vertices of R.

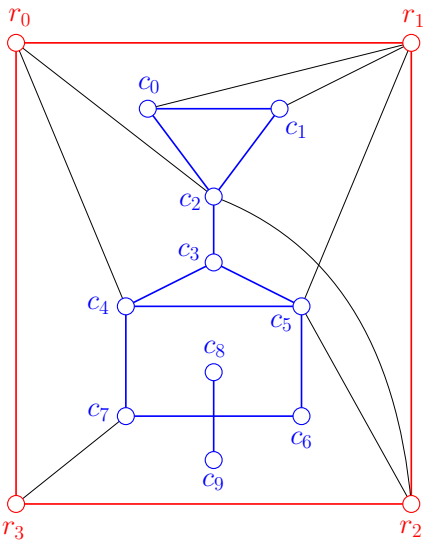

Figure 2.1: Graph of the game in Example 2.1
with the protected region consisting of 2 connected components.

The robber player places his unique robber pawn, called the robber, on a vertex in R to start the game. The cop player place his cop pawns, called cops, on vertices in C and can move them along the edges within the protected region $G[C]$. The robber can move either along the edges of the cycle $G[R]$ or along any edge connecting a vertex in the robber region to one in the protected region.

Since the second connected component of the protected region (consisting of vertices c_8, c_9) has no edge to any vertex in the robber region, the robber can not attack this component. Hence we do not need any cop to guard this component, and can omit the component completely from the graph.

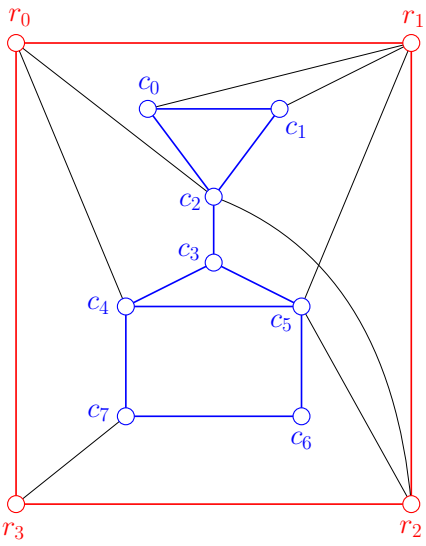

Figure 2.2: Graph of the game in Example 2.1 after omitting the connected component that is not adjacent to the robber region.

Figure 2.2 illustrated the graph of the game after removing this connected component.

From vertex r_1, the robber can move to one of the following vertices: r_0, r_2 in the robber region, c_0, c_1, c_5 in the protected region. Hence, when the robber stays on vertex r_1, there must be at least 3 cops staying on c_0, c_1, c_5. It implies that the cop player must use at least 3 cops to win the game. □

2.2 Modeling

In [5] Nagamochi modeled the problem stated in the previous section as an optimization problem on an auxiliary graph. In this section, we present the construction of that model in detail. Section 2.2.1 shows that the cop player needs to focus only on a special strategy, called *cyclic strategy*, of the robber player. Section 2.2.2 shows that the cop player only needs to construct a so-called *periodic strategy* to win against the cyclic strategy of the robber player. On that basis, Section 2.2.3 gives the detail construction of the auxiliary graph in the model of Nagamochi, and translates our considering problem to a path covering problem on the auxiliary graph.

2.2.1 Cyclic strategy

As mentioned in the previous section, a strategy of the robber player is understood informally as a way that he moves the robber in his turns. For the ease of our discussion, we need a formal definition for the concept of strategy of the robber player. For that, let I be the set of rounds in the game. The set I can be finite (either when the robber player decides to stop playing or when the robber is caught, in this case we have $I = \{0, 1, \ldots, |I| - 1\}$). The set I can be also infinite (when the robber player decides to continue playing, in this case we have $I = \mathbb{N}$).

Definition 2.2. (Strategy of the robber player, see [5]). *A strategy of the robber player is a sequence $\sigma = (r_{j_i})_{i \in I}$ of vertices in the robber region $G[R]$, in which:*

- *the robber is placed on the vertex $r_{j_i} \in R$ at round i of the game, and*
- *any two consecutive vertices in the sequence σ are adjacent in the robber region (i.e., r_{j_i} and $r_{j_{i+1}}$ are adjacent in $G[R]$).*

For instance, for the game in Example 2.1, $\sigma = (r_0, r_1, r_2, r_1, r_2, r_3, r_2)$ is a strategy of the robber player.

Recall that $G[R]$ is the cycle $r_0 - r_1 - \ldots - r_{|R|-1} - r_0$ and the robber can only move within the robber region before entering the protected region. These two facts suggest us to consider the following special strategy of the robber player.

- In the first turn, the robber player places the robber on r_0.
- In the next turns, the robber player moves the robber to r_1, then r_2, ..., and keep moving along the cycle $G[R]$ in the same direction.

This strategy is called *cyclic strategy* and has the following precise formulation.

Definition 2.3. (Cyclic strategy, see [5]). *The cyclic strategy of the robber is the sequence $\sigma^* = (r_{j_i})_{i \in I}$ in which*

$$j_i = i \bmod |R|. \tag{2.1}$$

For illustration, for the game in Example 2.1, the cyclic strategy of the robber player is

$$\sigma^* = (r_0, r_1, r_2, r_3, r_0, r_1, r_2, r_3, r_0, r_1, r_2, \ldots).$$

It is shown in the next section that the cop player will win the game if he can find a strategy against the cyclic strategy successfully.

2.2.2 Periodic strategy

Similar to the robber, it is understood informally that a strategy of the cop player is a way of placing the cops in each round of the game. For convenience, we need a formal definition for this concept. To do that, let us number the cops from 1 to n_c. The positions of these cops in round i of the game can be encoded by a vector $X_i = (c_1^i, c_2^i, \ldots, c_{n_c}^i)$, in which $c_k^i \in C$ is the vertex where the k^{th} cop is placed in this round. Note that more than one cop can be placed on a vertex in the protected region, therefore the vertices $c_1^i, c_2^i, \ldots, c_{n_c}^i$ are not necessarily pairwise distinct.

Definition 2.4. (Strategy of the cop player, see [5]). *A strategy of the cop player is a sequence $\pi = (X_i)_{i\in I}$, in which X_i is the vector of positions of cops in round i of the game.*

To be valid, a strategy of the cop player must adapt the game rules and the structure of the graph G. More precisely, in a valid strategy

$$\pi = (X_i)_{i\in I} = (c_1^i, c_2^i, \ldots, c_{n_c}^i)_{i\in I},$$

of the cop player, for each $i \in I$ and $k = 1, \ldots, n_c$, the vertices c_k^i and c_k^{i+1} are either coincide or adjacent to each other. Indeed, these two vertices are respectively the positions of the k^{th} cop in round i and $i+1$. According to the game rules, in each turn the cop can either stay at his current position or move to an adjacent vertex in the protected region. The former option corresponds to the case c_k^i and c_k^{i+1} are coincide, while the latter option corresponds to the case c_k^i and c_k^{i+1} are adjacent to each other.

The goal of the cop player is to win the game. Therefore, apart from being valid, his strategy must against the strategy of the robber player in the following sense.

Definition 2.5. (see [5]). *Let $\sigma = (r_{j_i})_{i\in I}$ be a strategy of the robber player. A strategy $\pi = (X_i)_{i\in I}$ of the cop player is called against σ if*

- *it is valid, and*
- *for each $i \in I$, the position vector X_i contains all vertices in C that are adjacent to r_{j_i}.*

The latter condition in the above means the following: whenever the robber stays on the vertex $r_{j_i} \in R$, each vertex in the protected region that is adjacent to r_{j_i} must be guarded by at least one cop. Roughly speaking, a strategy of

the cop player against a strategy σ is a way of placing the cops in each round so that the robber cannot come from any vertex of σ to the protected region without being caught.

Example 2.6. Consider the strategy $\sigma = (r_0, r_1, r_2, r_3)$ of the robber player in the guarding game played on the graph in Figure 2.2. With 4 cops, the cop player has strategy $\pi = (X_0, X_1, X_2, X_3)$ against σ, in which

$$X_0 = (c_0, c_2, c_3, c_4),$$
$$X_1 = (c_0, c_1, c_5, c_4),$$
$$X_2 = (c_0, c_2, c_5, c_4),$$
$$X_3 = (c_0, c_2, c_5, c_7).$$

We illustrate the strategies σ and π in Figure 2.3, in which the positions of the robber in his turns are represented by circles filled with red color, and the positions of the cops are represented respectively by circles filled with blue, green, purple, and orange colors.

The first position of the robber in the strategy σ is at vertex r_0, from which he can attack vertices c_2 and c_4 in the cop region. By placing 4 cops on components of X_0 (including c_2 and c_4, see Figure 2.3 (a)), the cop player prevents the protected region from the attack from position r_0 of the robber.

The next round is illustrated in Figure 2.3 (b). In this round, the robber moves from r_0 to r_1. The cops on vertices c_0 and c_4 stay there, while the cop on c_2 moves to c_1 and the one on c_3 moves to c_5. This forms the vector of positions $X_1 = (c_0, c_1, c_5, c_4)$. From r_1, the robber can attack the vertices c_0, c_1, c_5 in the cop region. Since these three vertices are included in X_1, the robber cannot move from r_1 to the protected region without being caught.

According to the strategy σ, in the third round the robber moves from r_1 to r_2. According to the strategy π, in this round the cops on vertices c_0, c_4, c_5 stay there, while the one on c_1 comes back to c_2 to form the vector of positions $X_2 = (c_0, c_2, c_5, c_4)$. This is illustrated in Figure 2.3 (c). In this way, the robber cannot safely move from r_2 to vertices c_2, c_5 in the protected region.

The last round is illustrated in Figure 2.3 (d). There, the robber moves from r_2 to r_3, and we need a cop to guard the vertex c_7. This can be done easily by moving the cop from c_4 in the previous round to c_7, and keep the positions of the other cops. We obtain the vector of positions $X_3 = (c_0, c_2, c_5, c_7)$ of the cops, which guard the protected region safely in this round. □

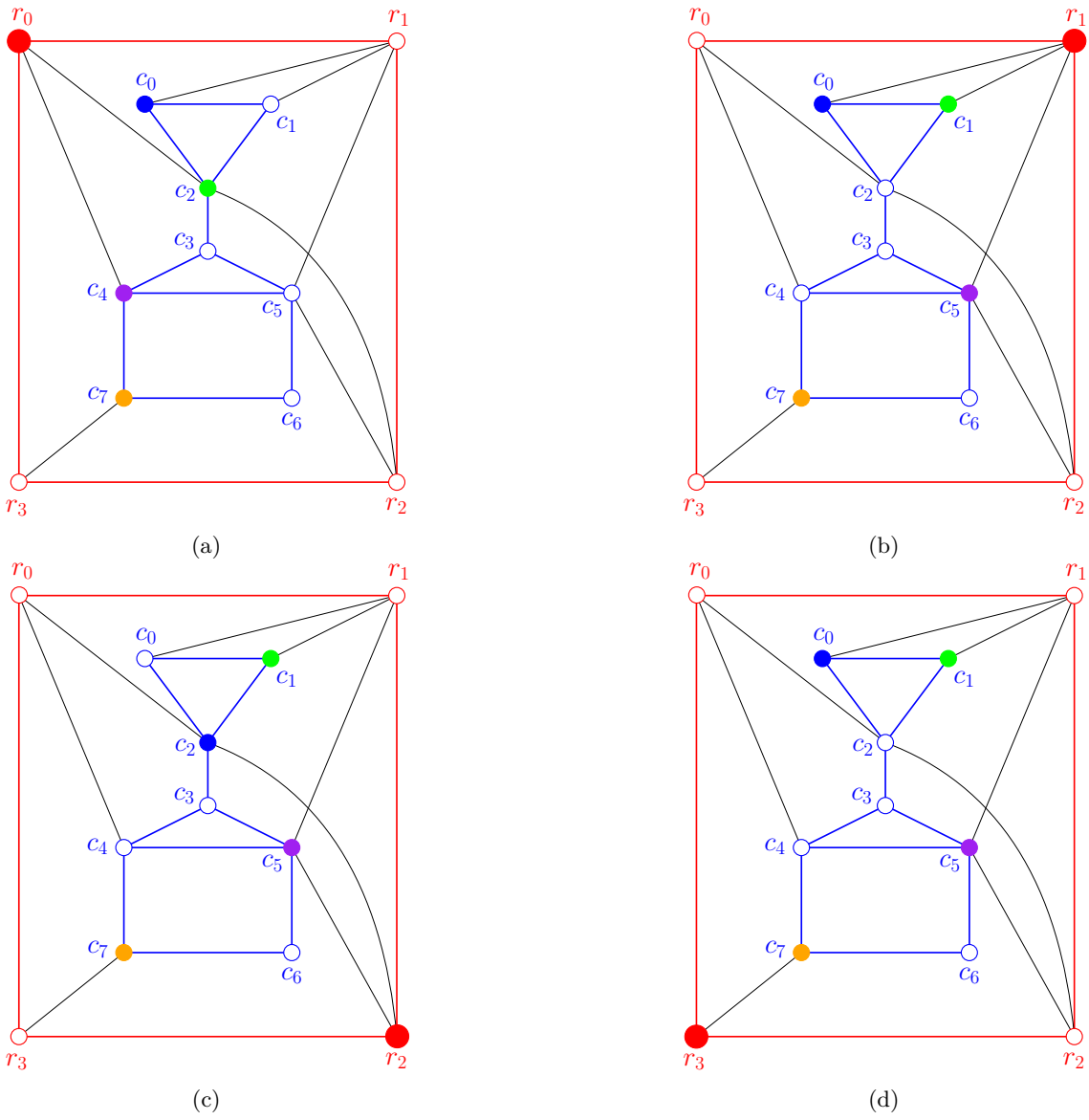

Figure 2.3: Illustration of strategy π in Example 2.6.

We now focus on how to find a strategy for the cop player to against the cyclic strategy $\sigma^* = (r_{j_i})_{i\in I}$ of the robber player as defined by (2.1). In the sequel, if not state differently, we only consider the case that $I = \mathbb{N}$.

Definition 2.7. (Periodic strategy of the cop player). *Let σ^* be the cyclic strategy of the robber. A strategy $\pi = (X_i)_{i\in I}$ of the cop player is called a periodic strategy if*

- *it against the cyclic strategy σ^*, and*

- *there exists a positive integer T such that for every $i \in I$ we have X_i is a permutation of components of $X_{i \bmod T|R|}$.*

A periodic strategy with $T = 1$ is called a compact strategy (of the cop player).

Roughly speaking, in a periodic strategy of the cop player, the cops return to their positions after each $T|R|$ rounds. In a compact strategy of the cop player, the cops return to their positions after each time the robber completes the cycle $G[R]$ of his region.

Example 2.8. Consider again the cop-robber guarding game played on the graph in Figure 2.2. The cyclic strategy of the robber in this game is

$$\sigma^* = (r_{j_i})_{i \in I}$$

in which $j_i = i \bmod |R| = i \bmod 4$. Let

$$\pi^* = (X_i)_{i \in I}$$

in which $X_i = X_{i \bmod 4}$ and X_0, X_1, X_2, X_3 are determined as in Example 2.6. The strategy π^* is a compact strategy of the cop player. Indeed, when the robber moves along the path $r_0 - r_1 - r_2 - r_3$, the cops move respectively according to the positions encoded by vectors X_0, X_1, X_2, X_3. When the robber moves from r_3 to r_0 to complete one time of going along the cycle $G[R]$, the cops on c_0 and c_2 stay there, while the cops on c_5 and c_7 respectively move to c_3 and c_4. In this way, when the robber completes the cycle $G[R]$ and returns to r_0, the cops also return to their initial positions encoded by X_0.

The main result in this subsection is the following lemma. It explains why the cop player only needs to focus on finding a periodic strategy to against the cyclic strategy of the robber player.

Lemma 2.9. (see [5], Lemma 2). *(i) If the cop player has a valid strategy with n_c cops against the cyclic strategy, then he also has a periodic strategy with the same number of cops against the cyclic strategy.*

(ii) Assume that the cop player has a periodic strategy with n_c cops against the cyclic strategy. Then, for any strategy σ of the robber player, the cop player can construct a valid strategy with the same number of cops against σ.

Proof. (i) Recall that the cyclic strategy of the robber player is $\sigma^* = (r_{j_i})_{i \in I}$ determined by (2.1). Let $\pi = (X_i)_{i \in I}$ be a strategy (not necessarily periodic) of the cop player with n_c cops against σ^*. According to the strategies σ^* and

π, in round i of the game, the robber is placed on vertex r_{j_i} and the cops are placed on components of $X_i = (c_1^i, c_2^i, \ldots, c_{n_c}^i)$. We therefore call

$$\alpha_i = (r_{j_i}, X_i) = (r_{j_i}, c_1^i, c_2^i, \ldots, c_{n_c}^i)$$

the state vector of round i.

Since the robber region $G[R]$ has $|R|$ vertices, there are $|R|$ choices for the first component of the state vector α_i. Since the protected region $G[C]$ has $|C|$ vertices and we can place more than one cop on each of the vertices, there are $|C|^{n_c}$ possible values for the position vector X_i. In total, there are at most $|R||C|^{n_c}$ different values for the state vector α_i. Therefore, from round 0 to round $|R||C|^{n_c}$, there must be two rounds having the same state vector, i.e., there exists two integers i_1, i_2 with $0 \leq i_1 < i_2 \leq |R||C|^{n_c}$ such that $\alpha_{i_1} = \alpha_{i_2}$.

Since the rounds i_1 and i_2 have the same state vector, the robber has the same position in these two rounds. According to the cyclic strategy, it means that from round i_1 to round i_2 the robber has completed $T = \frac{i_2 - i_1}{|R|}$ times of going through the whole cycle $G[R]$, in which T obtains a positive integral value.

Since $X_{i_1} \equiv X_{i_2}$, from round i_2 the cop player can repeat the strategy $(X_{i_1}, X_{i_1+1}, \ldots, X_{i_2-1})$ for each T times that the robber goes through the whole cycle $G[R]$. Formally, the cop player can follow a new strategy $\pi' = (X'_i)_{i \in I}$ in which

- $X'_i = X_i$ for $i = 0, \ldots, i_2$, and
- $X'_{i+kT|R|} = X'_i$ for $i = i_1, \ldots, i_2 - 1$ and $k = 1, 2, \ldots$

In that way, the cop player continues guarding successfully the protected region as he has done from round i_1 to round i_2.

Let i' be the smallest multiplication of $|R|$ satisfying $i_1 < i' \leq i_2$. As we have constructed, in the strategy π', the sequence

$$(X'_{i_1}, X'_{i_1+1}, \ldots, X'_{i_2-1})$$

is consecutively repeated from round i_1. Thus, the sequence

$$(X'_{i'}, X'_{i'+1}, \ldots, X'_{i_2-1}, X'_{i_2}, \ldots, X'_{i'+T|R|-1}) \tag{2.2}$$

is also consecutively repeated in the strategy π'. Note that by definition of i' we have $i' \bmod |R| = 0$, so in round i' the robber is on the first vertex r_0 of the robber region. Hence, by consecutively repeating the sequence (2.2)

from round 0, the cop player obtains a periodic strategy π^* against the cyclic strategy of the robber player.

(ii) We now assume that the cop player has a periodic strategy $\pi^* = (X_i^*)_{i\in I}$ with n_c cops against the cyclic strategy σ^*. Let σ be an arbitrary strategy of the robber player. In the following we will construct a strategy π_σ that also uses n_c cops to win against the strategy σ.

We say that the robber moves clockwisely if

- he leaves a vertex r_j with $0 \leq j < |R| - 1$ to move to the vertex r_{j+1}, or
- he leaves the vertex $r_{|R|-1}$ to move to r_0.

Similarly, we say that the robber moves counterclockwisely if

- he leaves a vertex r_j with $0 < j \leq |R| - 1$ to move to the vertex r_{j-1}, or
- he leaves the vertex r_0 to move to $r_{|R|-1}$.

If the strategy σ of the robber player starts from a vertex r_j rather than r_0, then we add the sequence $(r_0, r_1, \ldots, r_j)$ before σ. Correspondingly, the cop player starts his strategy π_σ by following the sequence $(X_0^*, X_1^*, \ldots, X_j^*)$ in the first $j+1$ rounds. In these rounds, the cops guard the protected region safely because $(r_0, r_1, \ldots, r_j)$ is a part of the cyclic strategy σ^* and the sequence $(X_0^*, X_1^*, \ldots, X_j^*)$ is the corresponding part of the periodic strategy π^* against σ^*. Hence, without loss of generality, we can assume that in the strategy σ the robber starts from the vertex r_0.

To win against the strategy σ, the cop player can follow the strategy π_σ constructed as follows.

- In the first round $j = 0$, when the robber is placed on r_0, the cops are placed on the components of X_0^*.
- Whenever the robber moves clockwisely from r_j to r_{j+1} (for any j in $\{0, 1, \ldots, |R| - 1\}$), the cops move from positions encoded in X_j^* to the ones in X_{j+1}^*.
- Whenever the robber moves clockwisely from $r_{|R|}$ to r_0, the cops move from positions encoded in $X_{|R|}^*$ to the ones in X_0^*.
- Whenever the robber moves counterclockwisely from r_j to r_{j-1} (for any j in $\{1, \ldots, |R|\}$), the cops move from positions encoded in X_j^* to the ones in X_{j-1}^*.

- Whenever the robber moves counterclockwisely from r_0 to $r_{|R|}$, the cops move from positions encoded in X_0^* to the ones in $X_{|R|}^*$.
- If in some round the robber stays on his current vertex and does not move, then the cops also stay on their current vertices.

Since the protected region is undirected, the above movement policy of the cops are possible. Since $\pi^* = (X_i^*)_{i\in I}$ wins against σ^*, in each of the above cases, the vector of positions of the cops prevents the robber from entering the protected region. Hence π_σ wins against the strategy σ of the robber player. □

Thanks to Lemma 2.9, the minimum number of cops that the cop player needs to ensure his winning is also the minimum number of cops in a periodic strategy against the cyclic strategy. To this end, what the cop player needs to do now is to find **a periodic strategy using the minimum number of cops**. The next subsection gives us one step closer to find such a strategy.

2.2.3 Auxiliary graph

This subsection presents the construction of a so-called *auxiliary graph* H, that helps us having an intuition of movements of cops against the cyclic strategy of the robber player.

Corresponding to each round $i \in I$, we make a copy $(c_0^i, \ldots, c_{|C|-1}^i)$ of vertices in the protected region $G[C]$, in which c_k^i is the copy of the vertex $c_k \in C$ for $k = 0, \ldots, |C| - 1$. The vertex set of the auxiliary graph H is

$$V_H = \{c_k^i \mid i \in I, k = 0, \ldots, |C| - 1\} \tag{2.3}$$

consisting of the copies of C in all rounds of the game. The copy corresponding to round i is called the *layer* i of the auxiliary graph H. Note that $|I| = +\infty$, therefore this graph has infinite number of layers and vertices.

In round i of the game, according to the cyclic strategy, the robber is placed on the vertex r_{j_i} determined by (2.1). Let $C_D^i \subset C$ be the set of vertices in C that are adjacent to r_{j_i}. We say that the vertices in C_D^i are *endangered*, since they can be attacked by the robber from r_{j_i} in one turn. In that spirit, for each $c_k \in C_D^i$, we also say that its copy c_k^i in layer i of the auxiliary graph is endangered.

We now consider an arbitrary cop and intend to represent his strategy by a directed path so-called *cop-path*

$$c_{k_0}^0 - c_{k_1}^1 - \ldots - c_{k_i}^i - \ldots \tag{2.4}$$

going through the layers of the auxiliary graph, in which each layer has exactly one vertex in this path. The vertex $c^i_{k_i}$ on the layer i belonging to this path means that the cop is placed on the vertex $c_{k_i} \in C$ in round i. Similarly, the vertex $c^{i+1}_{k_{i+1}}$ on the layer $i+1$ belonging to this path means that the cop is placed on the vertex $c_{k_{i+1}} \in C$ in round $i+1$. By the game rules, there are only two following possibilities for the cop when turning from round i to round $i+1$.

- *Possibility 1:* The cop stays on his current position, i.e., $c_{k_i} \equiv c_{k_{i+1}}$. In this case, $c^i_{k_i}$ and $c^{i+1}_{k_{i+1}}$ are two copies of the same vertex in the protected region, or simply we have $c^{i+1}_{k_{i+1}} = c^{i+1}_{k_i}$.
- *Possibility 2:* The cop moves to an adjacent vertex in the protected region, i.e., c_{k_i} and $c_{k_{i+1}}$ are adjacent in $G[C]$. This movement is represented by the arc from $c^i_{k_i}$ to $c^{i+1}_{k_{i+1}}$ in the path (2.4).

To guarantee Possibility 1 for all cops, we need arcs (c^i_k, c^{i+1}_k) for all indices $k = 0, \ldots, |C|-1$. To guarantee Possibility 2 for all cops, we need arcs (c^i_j, c^{i+1}_k) and (c^i_k, c^{i+1}_j) for each pair of adjacent vertices c_j, c_k in $G[C]$. The arc set A_H of the auxiliary graph H consists of these arcs for every round $i \in I$. More precisely, we have

$$\begin{aligned} A_H = &\{(c^i_k, c^{i+1}_k) \mid i \in I, k \in \{0, \ldots, |C|-1\}\} \\ &\cup \{(c^i_j, c^{i+1}_k), (c^i_k, c^{i+1}_j) \mid i \in I, j, k \in \{0, \ldots, |C|-1\}, \{c_j, c_k\} \in E(C)\}. \end{aligned} \quad (2.5)$$

The following proposition gives us some important properties of the auxiliary graph.

Proposition 2.10. *The auxiliary graph H with the vertex set V_H defined by (2.3) and the arc set A_H defined by (2.5) is an acyclic directed graph, and the vertices on the same layer of this graph are not reachable to each other.*

Proof. The arc set A_H certifies that H is a directed graph. By the construction of A_H, each arc in the auxiliary graph starts from a vertex in some layer $h \in I$ and ends at a vertex in the next layer $h+1$, and there is no arc in the reverse direction. Furthermore, there is no arc between vertices on the same layer. Thus, there does not exist any cycle in the graph, and the same-layer vertices are not reachable to each other. □

By definition, each vertex $c_k \in C^i_D$ is endangered in round i and therefore it needs at least one cop to be guarded. In other words, there must be at least one cop-path (representing the strategy of some cop) going through the

copy of the endangered vertex c_k in round i, i.e., through the vertex c_k^i in the layer i of the auxiliary graph. Roughly speaking, it means that the cop-paths must cover the endangered vertices in the auxiliary graph. Note that each cop corresponds to one cop-path, and each cop-path represents the strategy of one cop. Therefore, finding a strategy for the cop player against the cyclic strategy is equivalent to **finding the minimum number of cop-paths that cover the endangered vertices in the auxiliary graph**.

Example 2.11. Figure 2.4 illustrates the subgraph corresponding to the first 7 rounds of the auxiliary graph H for the game in Example 2.1. The vertices of H are arranged in 8 columns, each column consists of the copies of a vertex in the protected region. The vertices of H are also arranged in layers, each layer consists of the copies of all vertices of the protected region in a round. For simplicity, in Figure 2.4 we do not write down the name of each vertex, but the name of the layers and columns. The name of each vertex in the figure is determined by its layer and its column as follows: vertex c_k^h is on the layer corresponding to the round h and on the column corresponding to the vertex $c_k \in C$.

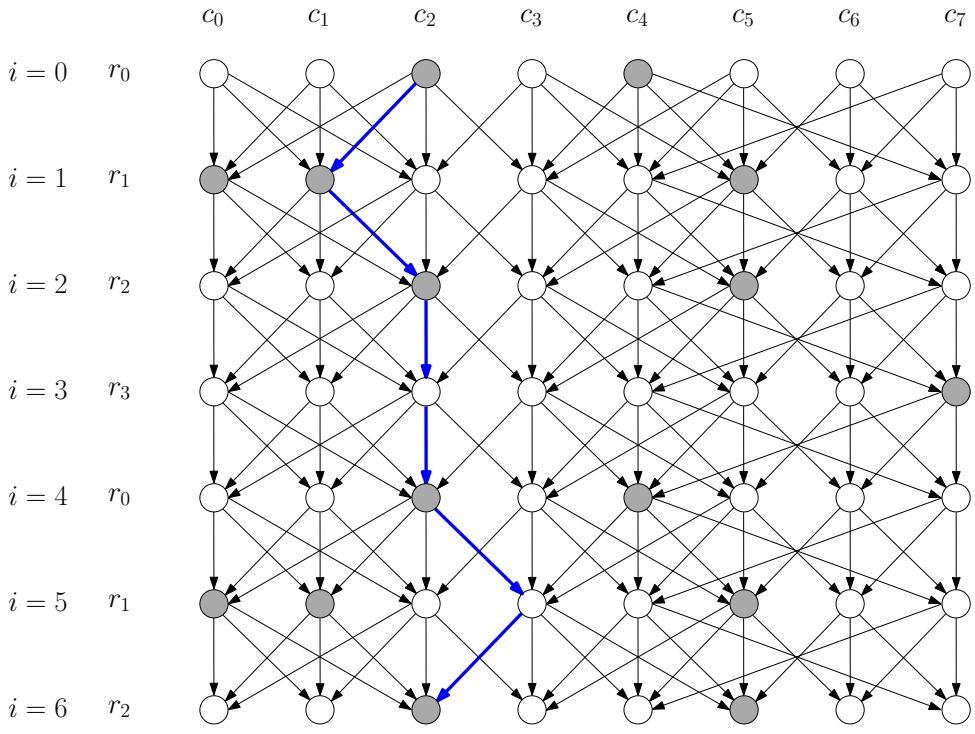

Figure 2.4: The part from round 0 to round 6 of the auxiliary graph for the game in Example 2.1.

Consider a round $h \in I$ of the game. The vertices in the layer h (resp., layer $h+1$) encode the status of the protected region in round h (resp., round $h+1$). The arcs from layer h to layer $h+1$ represent the possibilities of movements of the cops as follows.

- For each $k = 0, \ldots, 7$, we establish the arc (c_k^h, c_k^{h+1}) to ensure that: if a cop is placed on the vertex c_k in round h, then in the next round he can stay there.
- For each edge $\{c_k, c_\ell\}$ in the protected region, we establish the arc (c_k^h, c_ℓ^{h+1}) to ensure that: if a cop is placed on the vertex c_k in round h, then in the next round he can move to the vertex c_ℓ. We also establish the arc (c_ℓ^h, c_k^{h+1}) to guarantee that: if a cop is placed on the vertex c_ℓ in round h, then in the next round he can move to the vertex c_k. For instances, since c_4 and c_7 are adjacent to each other in the protected region, we establish the arcs (c_4^h, c_7^{h+1}) and (c_7^h, c_4^{h+1}). Since c_4 is not adjacent to c_6, we do not establish the arc from c_4^h to c_6^{h+1} and also do not establish the arc from c_6^h to c_4^{h+1}.

According to the cyclic strategy, the robber starts from the vertex r_0 and moves clockwisely along the cycle $r_0 - r_1 - r_2 - r_3 - r_0$. Thus, from round $i = 0$ to round $i = 6$, the positions of the robber are respectively $r_0 - r_1 - r_2 - r_3 - r_0 - r_1 - r_2$. As illustrated in Figure 2.4, the position of the robber in each of these rounds is noted in the corresponding layer. In round $i = 0$, the robber is placed on r_0, from which he can attack c_2 and c_4. Thus, in layer 0 of the auxiliary graph, the vertices c_2^0 and c_4^0 are endangered. Similar to the other positions of the robber, we obtain the endangered vertices that are highlighted in grey color in Figure 2.4.

The path consisting of blue arcs in Figure 2.4 is the part in the first 7 rounds of a cop-path. This represents the strategy described in Table 2.1 of a cop in these rounds.

Round	0	1	2	3	4	5	6
Position of the robber	r_0	r_1	r_2	r_3	r_0	r_1	r_2
Position of the cop	c_2	c_1	c_2	c_2	c_2	c_3	c_2

Table 2.1: Strategy of the cop corresponding to the blue path in Figure 2.4.

□

2.3 Best lower bound

Following the discussion in the previous section, this section shows that a good lower bound on the minimum number of cop-paths covering the endangered vertices in the auxiliary graph can be found in polynomial time.

Given $q \in I$, let $H[0,q]$ be the subgraph from round 0 to round q of the auxiliary graph H. It follows from the construction of V_H and A_H in (2.3)-(2.5), the vertex set of $H[0,q]$ is

$$V[0,q] = \{c_k^h \mid 0 \le h \le q, 0 \le k \le |C|-1\},$$

and the arc set of $H[0,q]$ is

$$\begin{aligned} E[0,q] = &\{(c_k^h, c_k^{h+1}) \mid 0 \le h \le q-1, 0 \le k \le |C|-1\} \\ &\cup \{(c_k^h, c_\ell^{h+1}) \mid 0 \le h \le q-1, 0 \le k < \ell \le |C|-1, \{c_k, c_\ell\} \in E(C)\} \\ &\cup \{(c_\ell^h, c_k^{h+1}) \mid 0 \le h \le q-1, 0 \le k < \ell \le |C|-1, \{c_k, c_\ell\} \in E(C)\}, \end{aligned}$$

in which $E(C)$ is the edge set of the protected region $G[C]$. Let $V_D[0,q]$ be the set of endangered vertices in $H[0,q]$. From the construction of the auxiliary graph we have

$$V_D[0,q] = \{c_k^h \in V[0,q] \mid 0 \le h \le q, \{r_{j_h}, c_k\} \in E(R,C)\},$$

where r_{j_h} is determined by (2.1) and $E(R,C)$ is the set of edges joining a vertex in R with a vertex in C. As an illustration, the graph in Figure 2.4 is nothing but $H[0,6]$ for the game in Example 2.1, in which $V_D[0,6]$ are vertices highlighted in grey.

The subgraph $H[0,q]$ has $q+1$ layers, each layer has $|C|$ vertices. Hence the number of vertices of $H[0,q]$ is $|V[0,q]| = (q+1)|C|$. Since $V_D[0,q] \subset V[0,q]$, there are finitely many endangered vertices in this subgraph. Hence, there exists a maximum antichain A^q in $V_D[0,q]$. Furthermore, by construction of the auxiliary graph H, for each $k = 0, \ldots, |C|-1$ the cop-path

$$c_k^0 - c_k^1 - \ldots - c_k^q \tag{2.6}$$

goes through the vertices on the same column in $H[0,q]$, which corresponds to the vertex $c_k \in C$. By definition, any pair of vertices in antichain A^q cannot belong to the same cop-path in $H[0,q]$. Thus, each cop-path of form (2.6) can go through at most one vertex in A^q. Since there are exactly $|C|$ cop-paths of form (2.6), we deduce that

$$|A^q| \le |C|, \quad \forall q \ge 0,$$

and hence there exists

$$a^* = \max_{q \geq 0} |A^q|. \tag{2.7}$$

We are going to show the following results concerning a^*.

- a^* is a lower bound on the minimum number of cop-paths covering the endangered vertices in the auxiliary graph.
- a^* can be computed in polynomial time.

In fact, by some additional arguments, Nagamochi in [5] proved that a^* is exactly the minimum number of cop-paths covering the endangered vertices in the auxiliary graph. Thus we can say that a^* is the best lower bound for the optimal number of cops.

The former result above is stated more precisely in the following lemma.

Lemma 2.12. (see [5], Lemma 3). *Let n_c^* be the minimum number of cops in a periodic strategy of the cop player against the cyclic strategy of the robber player. Then we have $n_c^* \geq a^*$.*

Proof. By (2.7), let $q^* \geq 0$ be such that $a^* = |A^{q^*}|$. Each vertex in A^{q^*} must be on some cop-path, and no pair of vertices in A^{q^*} can be on the same cop-path (since A^{q^*} is an antichain). Thus, the number of cop-paths to cover A^{q^*} must be at least $|A^{q^*}|$. Since each cop-path corresponds one-to-one with a cop, it follows that $n_c^* \geq |A^{q^*}| = a^*$. □

The latter result above concerns the concept of graph diameter, which is defined precisely as follows.

Definition 2.13. (see [8], page 14). *Given a connected undirected graph G.*

(i) The distance between two distinct vertices u, v in G is the length of a shortest u-v-path.

(ii) A geodesic is a shortest u-v-path for some distinct vertices u, v in G.

(iii) The diameter of G, denoted $d(G)$, is the length of any longest geodesic of G.

It is worth noting that a shortest path joining two given vertices in a connected undirected graph can be found in polynomial time by the well-known Dijkstra's algorithm (see e.g. [4], Chapter 7). For illustration, in the protected region $G[C]$ of the game on the graph in Figure 2.2, the distance between c_0 and c_5 is 3 since the shortest c_0-c_5-path is $c_0 - c_2 - c_3 - c_5$ having length 3, and the diameter of this protected region is 4 since any pair of vertices in this region can be joined by a path of length at most 4.

The following lemma is important to prove that the value of a^* can be computed in polynomial time. It shows that a^* can be determined in a finite subgraph of the auxiliary graph H, so we do not need to consider the whole auxiliary graph.

Lemma 2.14. (see [5], Lemma 4). *Let d be the maximum value of the diameters of the connected components of $G[C]$. Then $H[0, |R| + d - 2]$ has an antichain A with $|A| = a^*$.*

Proof. By (2.7), we can let Q be the set of all values of q such that $|A^q| = a^*$. For each $q \in Q$, since each cop-path of form (2.6) can only contain at most vertex in the antichain A^q, this antichain admits the representation

$$A^q = \{c_j^{h_j} \mid j \in J\},$$

for some $J \subseteq \{0, 1, \ldots, |C| - 1\}$. Among such antichains A^q with $q \in Q$, let A be the antichain having minimum value of $\sum_{j \in J} h_j$. Since $|A^q| = a^*$ for all $q \in Q$, we also have $|A| = a^*$. It is left to show that A is an antichain in $H[0, |R| + d - 2]$.

Indeed, let G' be a connected component in the protected region $G[C]$. Since A is chosen such that $\sum_{j \in J} h_j$ is minimized, the component G' must contain a vertex c_j such that $c_j^{h_j} \in A$ and $0 \leq h_j \leq |R| - 1$. By definition of parameter d, every vertex in G' can be reached from the vertex c_j by a path of length at most d. In other word, a cop on the vertex c_j in round h_j can come to any vertex in G' after at most d rounds. In the language of the auxiliary graph, this means the following: from the vertex $c_j^{h_j} \in A$ on both layer h_j and column c_j, we can reach to the vertex c_k^h on both column corresponding to a vertex $c_k \in G'$ and some layer h satisfying $h_j < h \leq h_j + d$. Furthermore, from c_k^h we can reach any vertex lying both on a subsequent layer after h and on the same column c_k. Therefore, from $c_j^{h_j}$ we can reach any vertex c_k^h that is both on a column $c_k \in G'$ and on a layer $h \geq h_j + d$. So any vertex in the antichain A that is different from the vertex $c_j^{h_j}$ must be on some layer h' satisfying $h' \leq h_j + d - 1$. Since $h_j \leq |R| - 1$, we have $h' \leq |R| + d - 2$. To the end, all vertices of A are on the layers having indices at most $|R| + d - 2$ in the auxiliary graph. This means that A is an antichain in $H[0, |R| + d - 2]$. □

It is known from Corollary 14.7b [4] that a minimum number of paths covering the vertex set of a directed acyclic $G = (V, A)$ can be found in $\mathcal{O}(|V||A|)$. Since this minimum number of paths equals the maximum size of an antichain in the graph, it follows immediately that a maximum antichain in the digraph

can be also found in $\mathcal{O}(|V||A|)$. Keeping this result and the above lemma in mind, we come up with the following result on complexity of computing a^*.

Theorem 2.15. (see [5]). *The value of a^* is equal to the size of a maximum antichain in $H[0, |R|+d-2]$, and such an antichain can be determined in time $\mathcal{O}((|R|+d)^2|C|(|C|+|E|))$.*

Proof. The antichain A constructed in the proof of Lemma 2.14 is a maximum antichain in $H[0, |R|+d-2]$ and has size $|A| = a^*$. The subgraph $H[0, |R|+d-2]$ has $|R|+d-1$ layers, each layer has $|C|$ vertices, hence $H[0, |R|+d-2]$ has $n_V^H = |C|(|R|+d-1)$ vertices. The number of arcs between two consecutive layers is $|C|+2|E(C)|$. The arcs in the auxiliary graph only connected vertices between consecutive layers, therefore the subgraph $H[0, |R|+d-2]$ has $n_E^H = (|C|+2|E(C)|)(|R|+d-2)$ arcs. By applying the Dilworth's theorem on $H[0, |R|+d-2]$, the maximum antichain A can be found in time

$$\begin{aligned}\mathcal{O}(n_V^H(n_V^H+n_E^H)) &= \mathcal{O}(|C|(|R|+d-1)(|C|+2|E(C)|)(|R|+d-2)) \\ &= \mathcal{O}((|R|+d)^2|C|(|C|+|E|)).\end{aligned}$$

□

Conclusions

In this thesis, we have studied two topics concerning the maximum antichain problem. The graphical version of this problem aims to find a vertex subset in a simple directed acyclic graph having as many vertices as possible such that there is no directed path connecting any two distinct vertices of the vertex subset.

The first topic, which is studied in Chapter 1, is Dilworth's theorem. The theorem states that any maximum antichain in a directed acyclic graph has the same size as any minimum path covering of the vertex set of the graph. We have presented two proofs for this theorem. The first proof is adapted from the one of Dantzig and Hoffman in [2], which bases on the well-known strong duality theorem in linear programming. The second proof is adapted from the one of Perles in [3], which used an elegant induction technique.

The second topic, which is studied in Chapter 2, is an application of the maximum antichain problem in studying a cob-robber guarding game on an undirected graph. In that game, the robber region induces a cycle, and the cops need to find a winning strategy using minimum number of cops. We have presented in detail the construction of Nagamochi in [5] to transform the problem of finding the minimum number of cops in a winning strategy for the cop player to the problem of finding a maximum antichain in an auxiliary graph. This transformation allows us to find the optimal number of cops in a polynomial time.

Bibliography

[1] R. P. Dilworth. A decomposition theorem for partially ordered sets. *Annals of Mathematics*, 51, 161–166, 1950.

[2] G. B. Dantzig and A. J. Hoffman. Dilworth's theorem on partially orders sets. Pages 207-214 in *Linear inequalities and related systems* (H. W. Kuhn and A. W. Tucker eds.), Princeton University Press, 1956.

[3] M. A. Perles. A proof of Dilworth's decomposition theorem for partially ordered sets. *Israel Journal of Mathematics*, 1:105–107, 1963.

[4] A. Schrijver. *Combinatorial Optimization: Polyhedra and Efficiency.* Springer, Berlin, 2003.

[5] H. Nagamochi. Cop-robber guarding game with cycle robber region. Pages 74-84 in *Frontiers in Algorithmics* (X. Deng, J. E. Hopcroft, and J. Xue eds.), Springer Berlin Heidelberg, 2009.

[6] H. B. Enderton. *Elements of Set Theory.* Academic Press, 1977.

[7] G. L. Nemhauser and L. A. Wolsey. *Integer and Combinatorial Optimization.* John Wiley & Sons, 1988.

[8] F. Harary. *Graph Theory.* Addison-Wesley, 1969.